Supplementary materials: Symbolic Representation for Any-to-Any Generative Tasks

In Appendix A, we provide comprehensive details regarding the generative tasks in Sec. A.1 and present a thorough elaboration of the user study methodology in Sec. A.2. In Appendix B, we conduct an extensive qualitative analysis comparing our approach with existing unified multi-modal frameworks, focusing on the quality of generated results. In Appendix C, we present additional experimental investigations, including a detailed comparison of computational efficiency versus human expert evaluation in Sec. C.1, and an in-depth analysis of examples and specific effects across three distinct syntax options in Sec. C.2. Finally, we examine the broader societal implications in Appendix D and discuss current limitations along with future research directions in Appendix E. Code and dataset are available at Jiaqi-Chen-00/Any-2-Any.

A. Experimental setup

A.1. Details on generative tasks

Our framework was evaluated across 12 distinct generative tasks collected from ComfyUI Examples [1], as detailed in Table 1. For Image2Mesh task, input images for mesh generation tasks were obtained from ComfyUI-3D-Pack [9], whereas other images were sourced from public repositories such as Vecteezy, Pexels, and Freepik. The evaluated tasks encompass a range of transformations, including:

- ***Inpainting***: The task of image inpainting involves filling in appropriate content in the erased regions of a given image to generate a complete and visually coherent output.
- ***Outpainting***: The image outpainting task extends the given image by generating a larger scene that seamlessly extends beyond the original boundaries while maintaining visual consistency.
- ***Novel View Synthesis***: Novel view synthesis task takes a single object image as input and generates images of the object from novel viewpoints by inferring 3D geometric relationships from the 2D input.
- ***Image merge***: The image merge task combines two landscape images to generate a new image that inherits the visual characteristics and features from both input images harmoniously.
- ***Merge model***: Merge model task blends different checkpoints for text-to-image generation models, enabling the creation of images that exhibit a combination of diverse visual styles and features.
- ***Image2Mesh***: Image2Mesh task involves creating a 3D mesh model that corresponds to the given input image, capturing its geometric structure.
- ***Multi-view image***: Given a single image, the multi-view task produces images of the same object from multiple viewpoints, offering comprehensive visual perspectives.
- ***Image2Video***: Image2Video task creates a video sequence that is semantically related to the input image, expanding the static visual content into a dynamic narrative.
- ***Text2Audio***: Text2Audio task enables the creation of music with specific styles based on the atmosphere and emotions conveyed in the textual description, facilitating text-guided music composition.
- ***Text2Image***: Text2Image task synthesizes high-resolution, photorealistic images with rich details and cinematic quality based on the provided textual descriptions.
- ***Text2Mesh***: Text2Mesh task creates 3D model mesh files that correspond to the given textual descriptions, translating language into three-dimensional geometric representations.
- ***Text2Video***: Text2Video task involves creating video clips that align with the content and narrative described in the given text, bringing the written concepts to life through moving visuals.

A.2. User study setup

We selected five evaluators from diverse academic and cultural backgrounds[1]. To ensure objectivity, we employed a double-blind evaluation method, ensuring that the evaluators were unaware of the model source for each result and that the presentation order was randomized. For each generative task, we conducted a one-on-one user study comparing our framework with all state-of-the-art unified multimodal frameworks (Show-o [12], SEED-X [3], LWM [7], Unified-IO 2 [8]). Detailed evaluation guidelines were provided, incorporating two ranking criteria. Evaluators assigned ranks

[1] All evaluators were compensated with a wage of at least 30 US dollars per hour, which is higher than the statutory rate.

Table 1. **Task description.** Each line includes representative task examples and input-output modality pair for the task.

ID	Category	Example of natural language instruction	Input type	Output type
1	Inpainting	You are given an image named 'yosemite_inpaint_example.png'. This image has had part of it erased, please inpaint a woman at the erased part to output a complete image called 'woman_inpainted'.	Image	Image
2	Outpainting	You are given a image named 'yosemite_outpaint_example.png'. Please outpaint the scenery of the given image and output a image called 'scene_outpainted'.	Image	Image
3	Image merge	Given two images, 'mountains.png' and 'sunset.png', extract their visual features. Then, combine the extracted visual features to generate an image that depicts a beautiful scene with features from both input images. Finally, save the generated image as a file named 'BeautifulScene'.	Image	Image
4	Novel view synthesis	You are given an image named 'marble_statue.jpg', please generate an image of the same object but from a different point of view. Save the output image as 'statue_different_view'	Image	Image
5	Merge model	Generate an image with bottles containing a galaxy-like visual effect. Please merge two different checkpoints. Save the generated image as 'galaxy_bottles'.	Text	Image
6	Image2Mesh	You are given an image named 'marble_statue.jpg'. Please generate its 3D mesh and save the mesh as 'marble_statue_mesh.obj'	Image	Mesh
7	Multi-view image	You are given an image named 'marble_statue_rgba.png'. Please generate its multi-view images. The generated images' filename prefix should be 'Comfyui'.	Image	Image
8	Image2Video	You are given an image named 'mountains.png'. Please create a 14 frame video of beautiful scenery from it.	Image	Video
9	Text2Audio	Generate an electronic dance music audio file inspired by a theme of 'heaven church.' Use an empty latent audio sample as the base, apply conditioning from a text description, and finally save the generated audio file as 'electronic_audio'.	Text	Audio
10	Text2Image	Generate a high-resolution, cinematic image of an anthropomorphic fox in a sci-fi spaceship, wearing a spacesuit, with dramatic lighting and detailed features. The style should be realistic, high quality, in 4k resolution.	Text	Image
11	Text2Mesh	Generate a 3D mesh of a anime girl with short skirt and daisy blue eyes and save the mesh as 'cute_girl.obj'.	Text	Mesh
12	Text2Video	Create a video of a cup of coffee being poured, but instead of coffee, a miniature galaxy swirls out, with stars and planets floating in the liquid.	Text	Video

from 1 (best) to n under each criterion.

- ❶ For ***text-result alignment*** assessment, evaluators were asked to assess if generated results faithfully represented all key elements specified in the input instructions (including scene composition, objects, styles and effects). The evaluators should consider whether any required elements were missing or if there were unintended additions.
- ❷ For ***result quality*** assessment, evaluators evaluated the overall visual quality independent of the instructions. They focused on technical aspects like image sharpness, consistency of style, composition balance, and level of detail. For video outputs, they also considered motion smoothness and temporal coherence.

To evaluate performance uniformly, we averaged the rankings for result quality and text-result alignment, with lower scores indicating better performance, with 1 being the best.

Instruction: *Given two images, 'beach.png' and 'space_nebula.png', extract their visual features. Then, combine the extracted visual features to generate an image of a beach with a surreal nebula sky. Finally, save the generated image as a file named 'SpaceBeach.png'.*

beach.png space_nebula.png SpaceBeach.png

Figure 1. **Example of image merge task.**

Table 2. **Average time cost (in seconds) compared to human ComfyUI experts**. "NVS" denotes "Novel View Synthesis", "I-2-3D" denotes "Image to 3D Mesh", "T2M" denotes "Text to Mesh".

Method	**Inpaint**	**Outpaint**	**Img merge**	**NVS**	**Merge model**	**I-2-3D**
Human	497.25	466.50	773.00	646.00	737.67	-
Ours	**62.00**	**29.60**	**79.70**	**37.50**	**40.80**	**117.30**
Speed up	8.02×	15.76×	9.70×	17.23×	18.08×	-
Method	**T2I**	**T2A**	**Multi-view img**	**I2V**	**T2M**	**T2V**
Human	537.25	278.00	-	590.00	-	1065.00
Ours	**36.10**	**41.80**	**43.40**	**116.50**	**155.60**	**72.00**
Speed up	14.88×	6.65×	-	5.06×	-	14.79×

B. Qualitative comparison

In Figure 2 to 13, we compare our method with mainstream unified models (Show-o [12], SEED-X [3], LWM [7], Unified-IO 2 [8], i-Code-V3 [11] and AnyGPT [14]) across different generation tasks. Due to our model's compositional nature, the LMs can invoke the most specialized functions and configure optimal parameters for each task. This flexibility in $\mathcal{A}$-language's functions and parameters enables superior task-specific adaptation compared to implicit neural representations, which use identical weights and frameworks across all tasks. Furthermore, our framework eliminates the need for multi-task trade-offs in design, resulting in performance levels comparable to single-task expert models - all achieved in a training-free setting.

C. Additional experiments

C.1. Time cost: Human experts *vs.* Ours

Setup To evaluate the efficiency of our proposed method, four human experts, proficient in ComfyUI workflow construction, were invited to build the 12 generative workflows from scratch. We conducted experiments comparing the average time cost (in seconds) required by these human experts and our method for the corresponding tasks.[2]

- ❶ The timing started from the moment the experts saw the task and ended when they produced a result that matched the instructions.
- ❷ We provided reference key functions and parameters, and allowed the experts to use any online tools. They were not allowed to directly copy existing workflows to the workspace, but had to construct their own workflows based on the reference information.
- ❸ The human experts were not required to optimize the workflows or improve the quality of the outcomes, they only needed to complete the tasks.

Results As shown in Table 2, compared to the human experts with over 1 years of experience, our method achieved an efficiency improvement of **5-18 times**.

- ❶ **Source of efficiency gains:** Human experts, regardless of their years of experience, inevitably require substantial time for task contemplation and workspace manipulation. This inherent time investment cannot be eliminated from their workflow. In contrast, our approach leverages pre-trained LMs to generate workflows instantaneously, completing the task within seconds and eliminating the cognitive overhead and manual ComfyUI interface operations. The primary time expenditure is workflow execution and subsequent refinement phases.
- ❷ **Fluctuations in time costs:** Human experts exhibit significant variations in task completion times, ranging from 278.00 seconds for Text2Audio to 1065.00 seconds for Text2Video tasks, primarily due to the varying complexity of problem-solving and debugging processes. In contrast, our LM-based inference approach maintains nearly constant cognitive processing time, with time variations primarily attributed to workflow execution and refinement phases. This results in substantially smaller fluctuations, spanning from 29.60 seconds for Outpainting to 155.00 seconds for Text2Mesh tasks.

C.2. Details comparison with different syntaxes

To illustrate the differences between the syntaxes of $\mathcal{A}$-Language, we use the Image merge task as an example. The input, output, and instruction for this task can be found in Figure 1, while examples of the three different syntaxes are presented in Sections C.2.1 to C.2.3.

In Table 3, we compare three syntax styles for $\mathcal{A}$-Language. The Declarative syntax demonstrates superior overall performance, achieving an average compile rate of 0.97 and an execute rate of 0.77.

[2]Since both human experts and the proposed method are based on the ComfyUI platform, their achieved results are not significantly different in quality. Therefore, it is not necessary to compare the quality differences.

C.2.1. Example of dataflow syntax

```
# create nodes by instantiation
clipvisionencode_13 = CLIPVisionEncode()
clipvisionencode_36 = CLIPVisionEncode()
emptylatentimage_5 = EmptyLatentImage(width=768, height=768, batch_size=1)
unclipcheckpointloader_32 = unCLIPCheckpointLoader(ckpt_name='sd21-unclip-h.ckpt')
ksampler_3 = KSampler(seed=947446491266673, control_after_generate='randomize', steps=26, cfg=8,
↪  sampler_name='uni_pc_bh2', scheduler='normal', denoise=1)
cliptextencode_6 = CLIPTextEncode(text='beach with a surreal nebula sky')
cliptextencode_7 = CLIPTextEncode(text='boring, drab')
unclipconditioning_19 = unCLIPConditioning(strength=0.5, noise_augmentation=0.4)
unclipconditioning_37 = unCLIPConditioning(strength=0.5, noise_augmentation=0.4)
loadimage_beach = LoadImage(image='beach.png')
loadimage_nebula = LoadImage(image='space_nebula.png')
vaedecode_8 = VAEDecode()
saveimage_result = SaveImage(filename_prefix='SpaceBeach')

# link nodes by invocation
model_32, clip_32, vae_32, clip_vision_32, name_string_32 = unclipcheckpointloader_32()
image_beach, mask_beach = loadimage_beach()
image_nebula, mask_nebula = loadimage_nebula()
clip_vision_output_13 = clipvisionencode_13(clip_vision=clip_vision_32, image=image_beach)
clip_vision_output_36 = clipvisionencode_36(clip_vision=clip_vision_32, image=image_nebula)
conditioning_6 = cliptextencode_6(clip=clip_32)
negative_conditioning_7 = cliptextencode_7(clip=clip_32)
conditioning_19 = unclipconditioning_19(conditioning=conditioning_6,
↪  clip_vision_output=clip_vision_output_13)
conditioning_37 = unclipconditioning_37(conditioning=conditioning_19,
↪  clip_vision_output=clip_vision_output_36)
latent_5 = emptylatentimage_5()
latent_3 = ksampler_3(model=model_32, positive=conditioning_37, negative=negative_conditioning_7,
↪  latent_image=latent_5)
image_8 = vaedecode_8(samples=latent_3, vae=vae_32)
result = saveimage_result(images=image_8)
```

C.2.2. Example of declarative syntax

```
# Add Node
workflow.add_node("clipvisionencode_13", "CLIPVisionEncode", {})
workflow.add_node("emptylatentimage_5", "EmptyLatentImage", {"width": 768, "height": 768,
↪  "batch_size": 1})
workflow.add_node("unclipcheckpointloader_32", "unCLIPCheckpointLoader", {"ckpt_name":
↪  'sd21-unclip-h.ckpt'})
workflow.add_node("clipvisionencode_36", "CLIPVisionEncode", {})
workflow.add_node("ksampler_3", "KSampler", {"seed": 947446491266673, "control_after_generate":
↪  'randomize', "steps": 26, "cfg": 8, "sampler_name": 'uni_pc_bh2', "scheduler": 'normal',
↪  "denoise": 1})
workflow.add_node("cliptextencode_6", "CLIPTextEncode", {"text": 'beautiful photograph'})
workflow.add_node("cliptextencode_7", "CLIPTextEncode", {"text": 'bad hands'})
workflow.add_node("unclipconditioning_19", "unCLIPConditioning", {"strength": 0.5,
↪  "noise_augmentation": 0.4000000000000002})
workflow.add_node("unclipconditioning_37", "unCLIPConditioning", {"strength": 0.5,
↪  "noise_augmentation": 0.4000000000000002})
workflow.add_node("loadimage_1", "LoadImage", {"image": 'beach.png'})
workflow.add_node("loadimage_2", "LoadImage", {"image": 'space_nebula.png'})
workflow.add_node("vaedecode_8", "VAEDecode", {})
workflow.add_node("saveimage_9", "SaveImage", {"filename_prefix": 'SpaceBeach'})
```

```
# Invoke Node
workflow.invoke_node(["image_1", "mask_1"], "loadimage_1")
workflow.invoke_node(["image_2", "mask_2"], "loadimage_2")
workflow.invoke_node(["model_32", "clip_32", "vae_32", "clip_vision_32", "name_string_32"],
↪  "unclipcheckpointloader_32")
workflow.invoke_node(["latent_5"], "emptylatentimage_5")
workflow.invoke_node(["clip_vision_output_13"], "clipvisionencode_13")
workflow.invoke_node(["clip_vision_output_36"], "clipvisionencode_36")
workflow.invoke_node(["conditioning_6"], "cliptextencode_6")
workflow.invoke_node(["conditioning_7"], "cliptextencode_7")
workflow.invoke_node(["conditioning_19"], "unclipconditioning_19")
workflow.invoke_node(["conditioning_37"], "unclipconditioning_37")
workflow.invoke_node(["latent_3"], "ksampler_3")
workflow.invoke_node(["image_8"], "vaedecode_8")

# Link Node
workflow.connect("clip_vision_32", "clipvisionencode_13", "clip_vision")
workflow.connect("image_1", "clipvisionencode_13", "image")
workflow.connect("clip_vision_32", "clipvisionencode_36", "clip_vision")
workflow.connect("image_2", "clipvisionencode_36", "image")
workflow.connect("clip_32", "cliptextencode_6", "clip")
workflow.connect("clip_32", "cliptextencode_7", "clip")
workflow.connect("conditioning_6", "unclipconditioning_19", "conditioning")
workflow.connect("clip_vision_output_13", "unclipconditioning_19", "clip_vision_output")
workflow.connect("conditioning_7", "unclipconditioning_37", "conditioning")
workflow.connect("clip_vision_output_36", "unclipconditioning_37", "clip_vision_output")
workflow.connect("conditioning_19", "ksampler_3", "positive")
workflow.connect("conditioning_37", "ksampler_3", "negative")
workflow.connect("model_32", "ksampler_3", "model")
workflow.connect("latent_5", "ksampler_3", "latent_image")
workflow.connect("latent_3", "vaedecode_8", "samples")
workflow.connect("vae_32", "vaedecode_8", "vae")
workflow.connect("image_8", "saveimage_9", "images")
```

C.2.3. Example of pseudo-natural syntax

```
# create nodes by instantiation
clipvisionencode_13 is CLIPVisionEncode()
clipvisionencode_36 is CLIPVisionEncode()
emptylatentimage_5 is EmptyLatentImage with the parameters of (width is 768, height is 768, batch_size
↪  is 1)
unclipcheckpointloader_32 is unCLIPCheckpointLoader with the parameters of (ckpt_name is
↪  'sd21-unclip-h.ckpt')
ksampler_3 is KSampler with the parameters of (seed is 947446491266673, control_after_generate is
↪  'randomize', steps is 26, cfg is 8, sampler_name is 'uni_pc_bh2', scheduler is 'normal', denoise
↪  is 1)
cliptextencode_6 is CLIPTextEncode with the parameters of (text is 'beach with a surreal nebula sky')
cliptextencode_7 is CLIPTextEncode with the parameters of (text is 'boring, drab')
unclipconditioning_19 is unCLIPConditioning with the parameters of (strength is 0.5,
↪  noise_augmentation is 0.4)
unclipconditioning_37 is unCLIPConditioning with the parameters of (strength is 0.5,
↪  noise_augmentation is 0.4)
loadimage_beach is LoadImage with the parameters of (image is 'beach.png')
loadimage_nebula is LoadImage with the parameters of (image is 'space_nebula.png')
vaedecode_8 is VAEDecode()
saveimage_result is SaveImage with the parameters of (filename_prefix is 'SpaceBeach')

# link nodes by invocation
model_32, clip_32, vae_32, clip_vision_32, name_string_32 is unclipcheckpointloader_32()
```

```
image_beach, mask_beach is loadimage_beach()
image_nebula, mask_nebula is loadimage_nebula()
clip_vision_output_13 is clipvisionencode_13 with the parameters of (clip_vision is clip_vision_32,
↪  image is image_beach)
clip_vision_output_36 is clipvisionencode_36 with the parameters of (clip_vision is clip_vision_32,
↪  image is image_nebula)
conditioning_6 is cliptextencode_6 with the parameters of (clip is clip_32)
negative_conditioning_7 is cliptextencode_7 with the parameters of (clip is clip_32)
conditioning_19 is unclipconditioning_19 with the parameters of (conditioning is conditioning_6,
↪  clip_vision_output is clip_vision_output_13)
conditioning_37 is unclipconditioning_37 with the parameters of (conditioning is conditioning_19,
↪  clip_vision_output is clip_vision_output_36)
latent_5 is emptylatentimage_5()
latent_3 is ksampler_3 with the parameters of (model is model_32, positive is conditioning_37,
↪  negative is negative_conditioning_7, latent_image is latent_5)
image_8 is vaedecode_8 with the parameters of (samples is latent_3, vae is vae_32)
result is saveimage_result with the parameters of (images is image_8)
```

D. Social impacts

While mainstream research focuses on larger single-weight models on leaderboards, real-world AI application practices [2, 10], particularly in the multimodal content generation domain, increasingly employ composite workflows with multiple components, especially those based on platforms like ComfyUI and Blender. This paper distills the essential elements of these composite workflows into a simple symbolic language, allowing pre-trained language models to directly infer the workflows. This modular approach enables developers to create AIGC workflows with minimal coding effort.

However, this approach has potential negative impacts on the AIGC field. Firstly, it could change the production methods of AIGC practitioners, leading to a shift in the modes of labor within the domain. Traditional AIGC workflows often require practitioners to manually combine and adjust individual components, whereas using symbolic languages and pre-trained models to infer workflows can automate this process, reducing reliance on manual operations. Secondly, as the degree of automation increases, the amount of labor required in the field may decrease, impacting employment to a certain extent.

Despite these concerns, adopting symbolic languages and pre-trained models to infer workflows still has significant advantages. It can lower the entry barrier for AIGC workflow development, allowing more people to participate in the field. Moreover, by automating some repetitive and time-consuming work, practitioners can focus more on creativity and optimization, improving production efficiency and content quality.

E. Limitation and future work

While this paper establishes the foundational concepts and definitions for the formal language and inference method, further research and development are needed to bridge the gap between the proposed approach and real-world applications. This may involve incorporating domain-specific knowledge, integrating with hierarchical designs [5], tree search techniques [6], and advanced agentic learning-based strategies [15], and addressing practical concerns such as computational efficiency, user experience, and system robustness.

References

[1] ComfyAnonymous. Comfyui examples. https://comfyanonymous.github.io/ComfyUI_examples/, 2023. 1

[2] Yilun Du and Leslie Kaelbling. Compositional generative modeling: A single model is not all you need. arXiv preprint arXiv:2402.01103, 2024. 7

[3] Yuying Ge, Sijie Zhao, Jinguo Zhu, Yixiao Ge, Kun Yi, Lin Song, Chen Li, Xiaohan Ding, and Ying Shan. Seed-x: Multimodal models with unified multi-granularity comprehension and generation. arXiv preprint arXiv:2404.14396, 2024. 1, 3

[4] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In CVPR, pages 14953–14962, 2023. 8

[5] Sirui Hong, Yizhang Lin, Bangbang Liu, Binhao Wu, Danyang Li, Jiaqi Chen, Jiayi Zhang, Jinlin Wang, Lingyao Zhang, Mingchen Zhuge, et al. Data interpreter: An llm agent for data science. arXiv preprint arXiv:2402.18679, 2024. 7

[6] Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents. arXiv preprint arXiv:2407.01476, 2024. 7

[7] Hao Liu, Wilson Yan, Matei Zaharia, and Pieter Abbeel. World model on million-length video and language with ringattention. arXiv preprint, 2024. 1, 3

[8] Jiasen Lu, Christopher Clark, Sangho Lee, Zichen Zhang, Savya Khosla, Ryan Marten, Derek Hoiem, and Aniruddha Kembhavi. Unified-io 2: Scaling autoregressive multimodal models with vision, language, audio, and action, 2023. 1, 3

[9] MrForExample. Comfyui-3d-pack. https://github.com/MrForExample/ComfyUI-3D-Pack, 2024. 1

[10] Lin Qiao. Fireworks ai raises 52m series b to lead industry shift to compound ai systems. https://fireworks.ai/blog/fireworks-ai-series-b-compound-ai, 2024. 7

[11] Zineng Tang, Ziyi Yang, Chenguang Zhu, Michael Zeng, and Mohit Bansal. Any-to-any generation via composable diffusion. arXiv preprint arXiv:2305.11846, 2023. 3

[12] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. arXiv preprint arXiv:2408.12528, 2024. 1, 3

[13] Xiangyuan Xue, Zeyu Lu, Di Huang, Wanli Ouyang, and Lei Bai. Genagent: Build collaborative ai systems with automated workflow generation–case studies on comfyui. arXiv preprint arXiv:2409.01392, 2024. 8

[14] Jun Zhan, Junqi Dai, Jiasheng Ye, Yunhua Zhou, Dong Zhang, Zhigeng Liu, Xin Zhang, Ruibin Yuan, Ge Zhang, Linyang Li, Hang Yan, Jie Fu, Tao Gui, Tianxiang Sun, Yugang Jiang, and Xipeng Qiu. Anygpt: Unified multimodal llm with discrete sequence modeling, 2024. 3

[15] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 19632–19642, 2024. 7

Table 3. **Performance comparison on syntax style.** We report the pass rate for a single run (Pass@1/%). "Comp" denotes "Compile", "Exec" denotes "Execute".

Syntax	Inpainting		Img merge		Outpainting		Novel view syn.		Merge model		Img2Mesh	
	Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec
Dataflow [4, 13]	**1.0**	**0.9**	**1.0**	0.7	0.9	0.9	0.7	0.5	**1.0**	**0.9**	**1.0**	**1.0**
Pseudo-natural	0.9	**0.9**	0.9	**0.8**	0.9	0.9	0.8	0.7	0.9	**0.9**	0.8	0.7
Declarative	**1.0**	0.7	0.9	0.7	**1.0**	**1.0**	**0.9**	**0.8**	0.9	**0.9**	**1.0**	0.8

Multi-view img		Img2Video		Text2Audio		Text2Img		Text2Mesh		Text2Video		Average	
Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec	Comp	Exec
0.8	0.1	**1.0**	0.9	0.6	0.4	**1.0**	0.9	0.3	0.3	**1.0**	**1.0**	0.86	0.71
0.9	0.4	**1.0**	**1.0**	0.6	0.4	0.8	0.8	0.3	0.0	0.9	0.8	0.81	0.63
1.0	**1.0**	**1.0**	**1.0**	**1.0**	**0.8**	**1.0**	**1.0**	**1.0**	**0.8**	**1.0**	0.9	**0.98**	**0.87**

Table 4. **Natural language instructions used in merge model for image generation.** The "position" column indicates the image's location in figure 9 grid using (`row`, `column`) coordinates, counting from top-left to bottom-right.

ID	Position	Natural Language Instruction
1	(1, 1)	Create a high-definition, futuristic image of a bustling neon-lit city at night, with towering skyscrapers, rain-soaked streets, and holographic billboards. The style should be cyberpunk, rich in vibrant colors and contrast. Please merge two different checkpoints.
2	(2, 1)	Produce an image of a Norse god standing atop a cliff with thunderous clouds and a glowing hammer. The scene should have dramatic, epic lighting in the style of classical oil paintings. Please merge two different checkpoints.
3	(3, 1)	Generate a highly detailed, 4K image of a steampunk inventor's workshop, filled with intricate gears, brass machines, and soft, warm lighting. The style should be vintage and richly textured. Please merge two different checkpoints.
4	(4, 1)	Create a beautiful underwater scene featuring a bioluminescent jellyfish forest with mythical creatures swimming around. The image should have a mystical, tranquil feel with soft blue-green hues and glowing details. Please merge two different checkpoints.
5	(5, 1)	Design a minimalist, high-contrast image of a lone cactus in a vast desert under a giant, crimson sun. The colors should be bold, with a surreal, almost abstract aesthetic. Please merge two different checkpoints.
6	(1, 2)	Produce a hauntingly beautiful portrait of a Victorian woman in dark attire, surrounded by a foggy, candlelit room with antique furniture. The style should be Gothic, with a moody, mysterious vibe. Please merge two different checkpoints.
7	(2, 2)	Generate a detailed illustration of an animal tea party in a forest clearing, featuring animals like rabbits and foxes dressed in Victorian attire. The style should be whimsical, with soft, pastel colors and charming details. Please merge two different checkpoints.
8	(3, 2)	Create a high-resolution image of an alien planet landscape with two suns and strange rock formations, set against a sky filled with vibrant galaxies. The style should be sci-fi with vibrant colors and atmospheric lighting. Please merge two different checkpoints.
9	(4, 2)	Produce an image of a retro-futuristic city with flying cars and curved glass buildings, all in a 1980s-inspired color palette. The style should be bold, with neon hues and a sense of nostalgic futurism. Please merge two different checkpoints.

Generate a high-resolution, cinematic image of an anthropomorphic fox in a sci-fi spaceship, wearing a spacesuit, with dramatic lighting and detailed features. The style should be realistic, high quality, in 4k resolution.

Create a high-definition, futuristic image of a bustling neon-lit city at night, with towering skyscrapers, rain-soaked streets, and holographic billboards. The style should be cyberpunk, rich in vibrant colors and contrast.

Produce an image of a Norse god standing atop a cliff with thunderous clouds and a glowing hammer. The scene should have dramatic, epic lighting in the style of classical oil paintings.

Generate a highly detailed, 4K image of a steampunk inventor's workshop, filled with intricate gears, brass machines, and soft, warm lighting. The style should be vintage and richly textured.

Create a beautiful underwater scene featuring a bioluminescent jellyfish forest with mythical creatures swimming around. The image should have a mystical, tranquil feel with soft blue-green hues and glowing details.

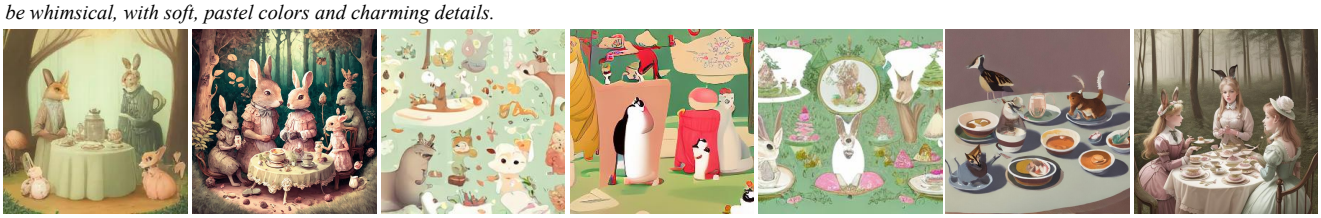
Show-o SEED-X LWM Unified-IO 2 i-Code-V3 AnyGPT Ours

Figure 2. **Qualitative results of Text2Image task (Part 1).**

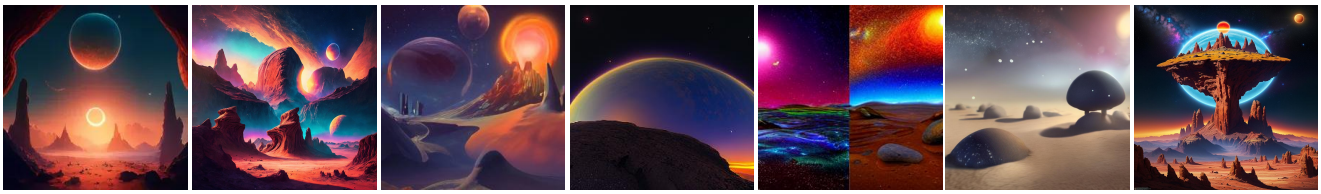
Design a minimalist, high-contrast image of a lone cactus in a vast desert under a giant, crimson sun. The colors should be bold, with a surreal, almost abstract aesthetic.

Produce a hauntingly beautiful portrait of a Victorian woman in dark attire, surrounded by a foggy, candlelit room with antique furniture. The style should be Gothic, with a moody, mysterious vibe.

Generate a detailed illustration of an animal tea party in a forest clearing, featuring animals like rabbits and foxes dressed in Victorian attire. The style should be whimsical, with soft, pastel colors and charming details.



Create a high-resolution image of an alien planet landscape with two suns and strange rock formations, set against a sky filled with vibrant galaxies. The style should be sci-fi with vibrant colors and atmospheric lighting.



Produce an image of a retro-futuristic city with flying cars and curved glass buildings, all in a 1980s-inspired color palette. The style should be bold, with neon hues and a sense of nostalgic futurism.

Show-o SEED-X LWM Unified-IO 2 i-Code-V3 AnyGPT Ours

Figure 3. **Qualitative results of Text2Image task (Part 2).**

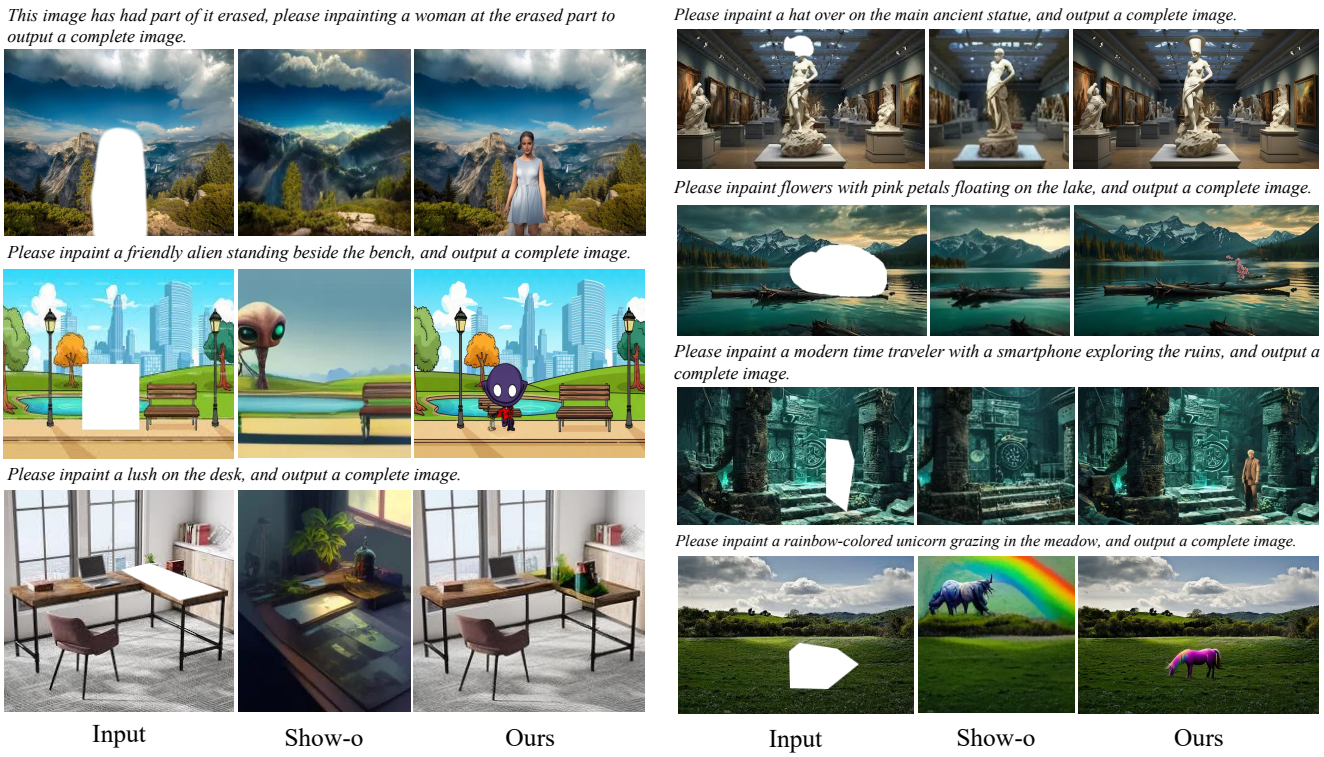


Figure 4. **Qualitative results of image inpainting.**

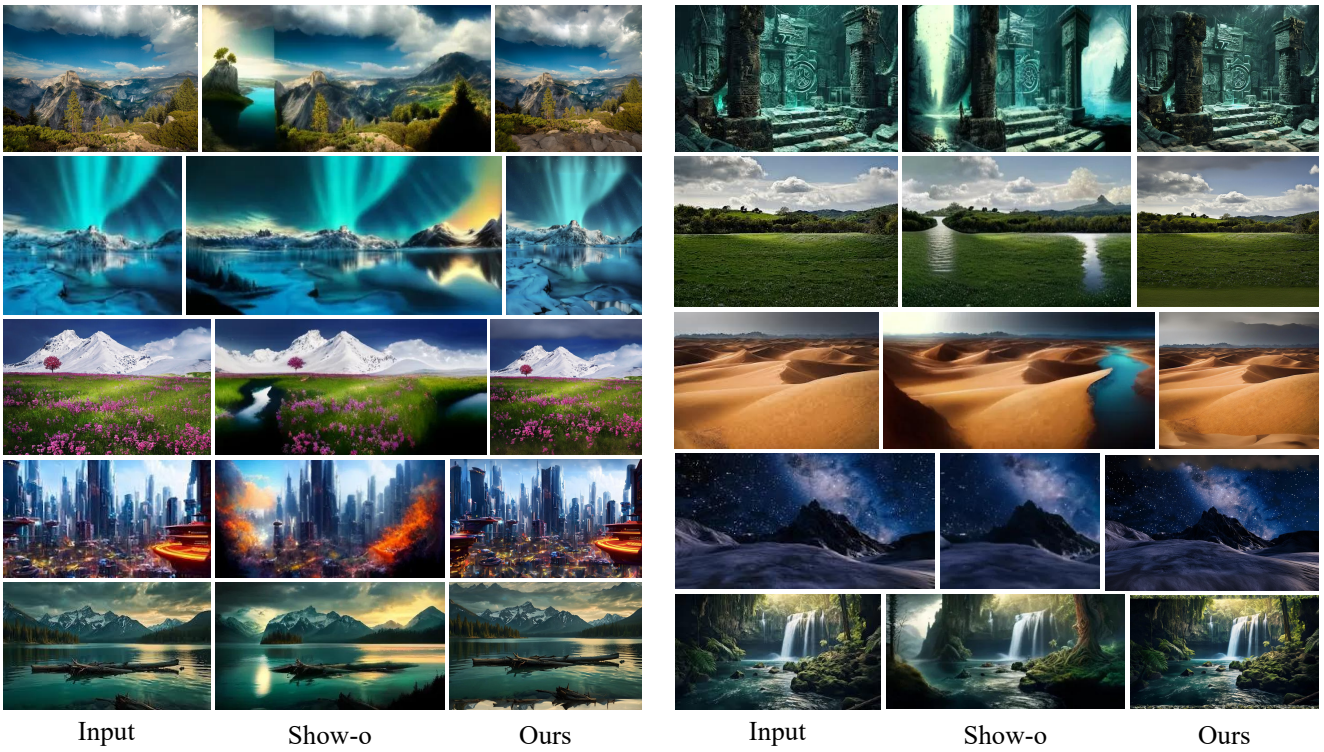


Figure 5. **Qualitative results of image outpainting.**

Figure 6. **Qualitative results of image merge.**

Figure 7. **Qualitative results of Text2Video.**

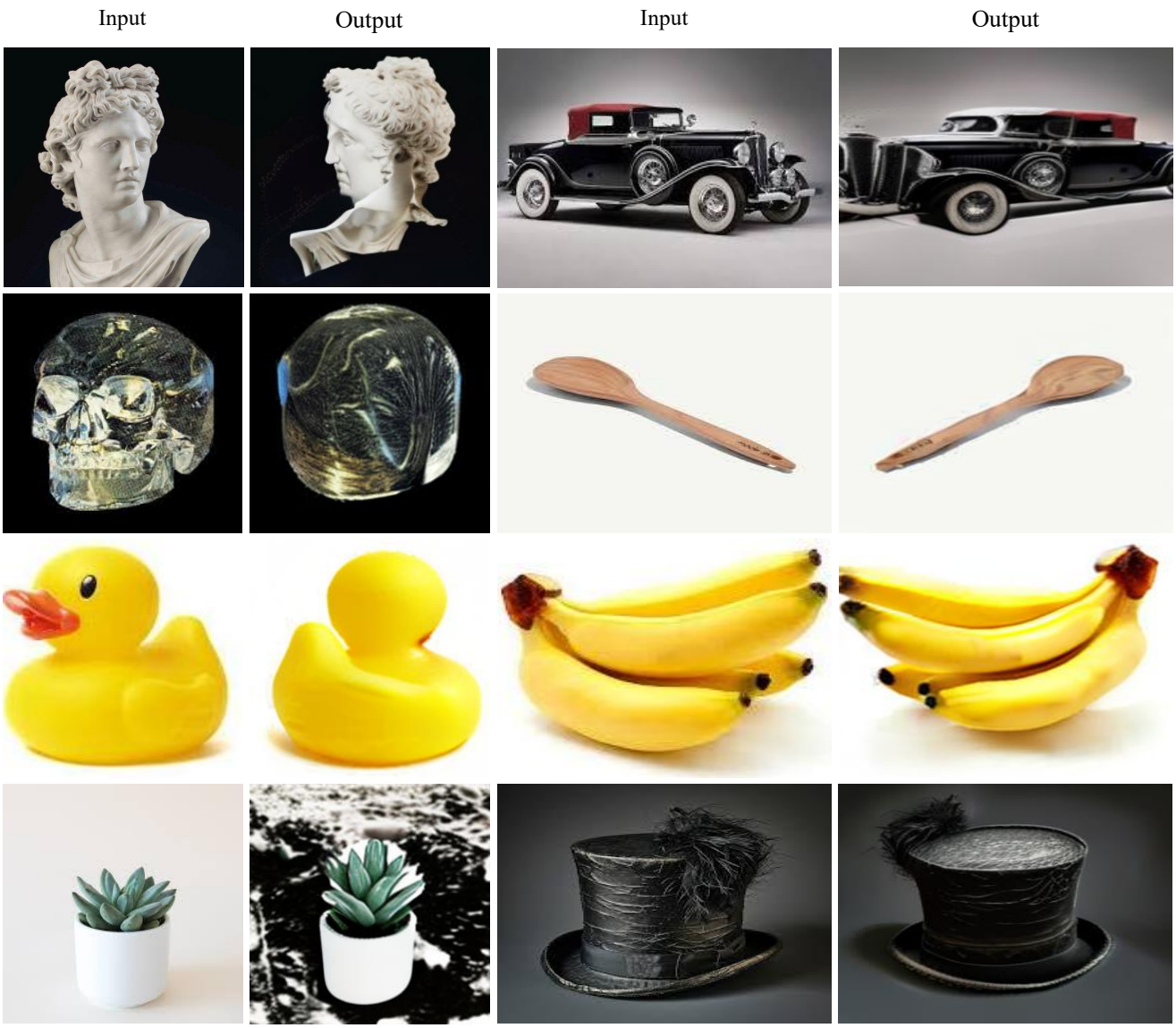


Figure 8. **Qualitative results of novel view synthesis (NVS).**

Figure 9. **Qualitative results of merge model.** This task allows visual style combinations through checkpoint blending. The visualization demonstrates generated outputs, where each image corresponds to its respective natural language instruction as detailed in Table 4.

Table 5. **Natural language instructions used in Text2Mesh generation.** The "position" column indicates the 3D mesh's location in figure 13 grid using (`row, column`) coordinates, counting from top-left to bottom-right.

ID	Position	Natural Language Instruction
1	(1, 1)	Generate a 3D mesh of a anime girl with short skirt and daisy blue eyes and save the mesh as 'cute_girl.obj'.
2	(2, 1)	Generate a 3D mesh of a plain ceramic coffee mug with a matte white finish. It features a gently curved, sturdy handle for gripping, a slightly rounded base, and a smooth, untextured surface that reflects faint ambient light. Save the mesh as 'coffee_mug.obj'.
3	(3, 1)	Generate a 3D mesh of a classic hardcover book with a solid blue cover. The book has a subtle fabric texture and rounded corners. The pages are aligned neatly with a slight golden tint at the edges, giving a vintage look. Save the mesh as 'blue_book.obj'.
4	(4, 1)	Generate a 3D mesh of a round, bright orange with a textured peel covered in small dimples. It has a tiny, dried green stem on top, and the surface shows a faint, shiny sheen, indicating juiciness. Save the mesh as 'orange_fruit.obj'.
5	(5, 1)	Generate a 3D mesh of a simple black office chair with a flat seat and a low backrest. It has a minimalistic design, thin matte finish, and stands on a five-wheel base, with each wheel small and unobtrusive. Save the mesh as 'office_chair.obj'.
6	(6, 1)	Generate a 3D mesh of a standard incandescent light bulb with a clear glass surface. The metallic base has grooved ridges for screwing in, and inside, a thin tungsten filament is suspended by two small metal wires. Save the mesh as 'light_bulb.obj'.
7	(1, 2)	Generate a 3D mesh of a simple wooden spoon with a smooth, polished light brown surface. The handle is straight with a slight curve at the end, and the bowl of the spoon is shallow with a rounded edge. Save the mesh as 'wooden_spoon.obj'.
8	(2, 2)	Generate a 3D mesh of a cylindrical transparent water bottle with a faint blue tint. It features a screw-on cap with ridges for grip, smooth body with slight indentations for holding, and tiny air bubbles trapped inside the water. Save the mesh as 'water_bottle.obj'.
9	(3, 2)	Generate a 3D mesh of a small green apple with a shiny, waxy surface. It has a slightly irregular shape, a tiny brown stem, and a smooth skin with light speckles and green highlights. Save the mesh as 'green_apple.obj'.
10	(4, 2)	Generate a 3D mesh of a traditional wooden pencil with a yellow hexagonal body. The pencil has a sharp graphite tip and a pink eraser on the other end, held by a shiny metal band. There are faint lines showing the wood grain pattern. Save the mesh as 'yellow_pencil.obj'.

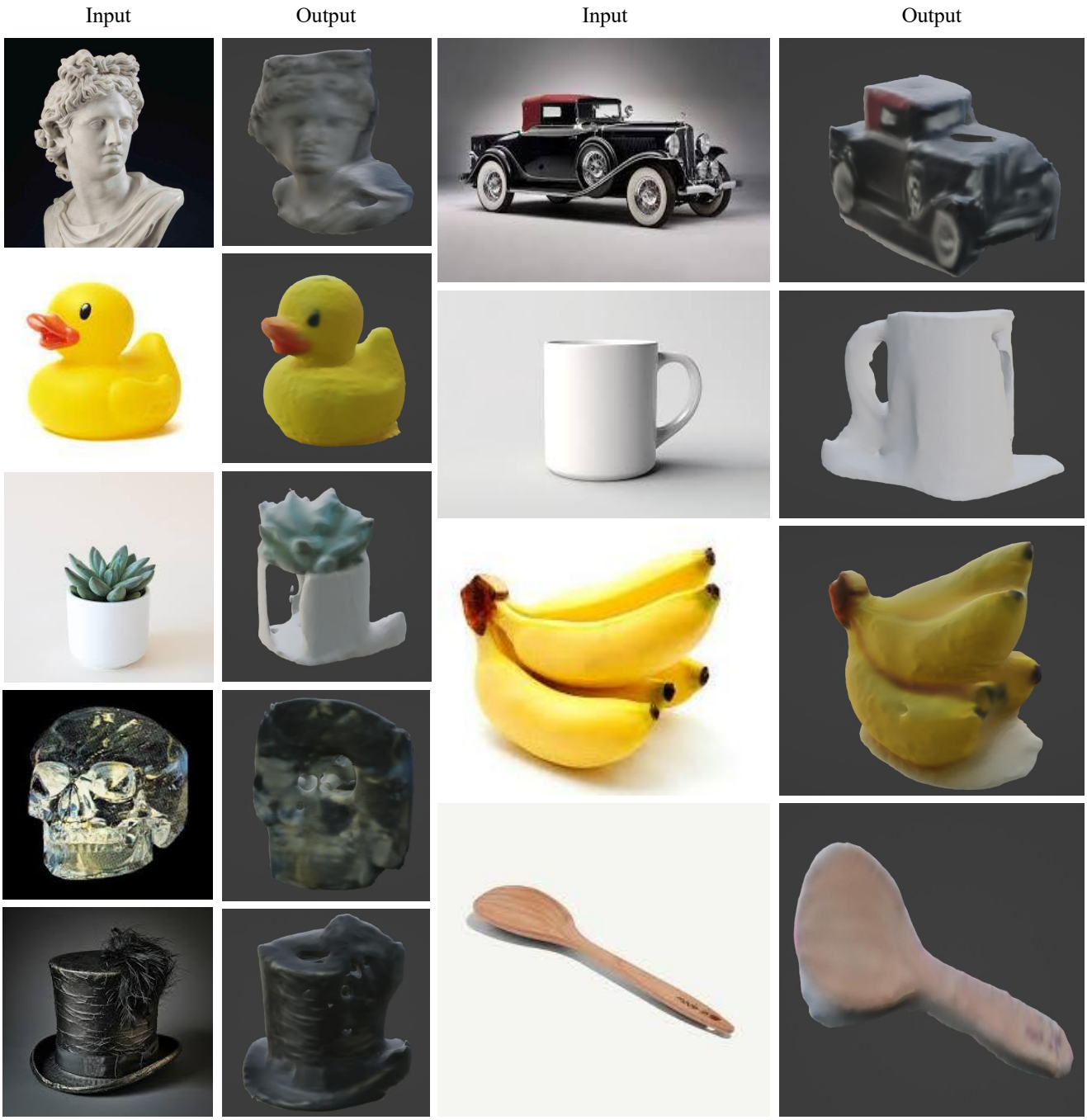


Figure 10. **Qualitative results of Image2Mesh.**

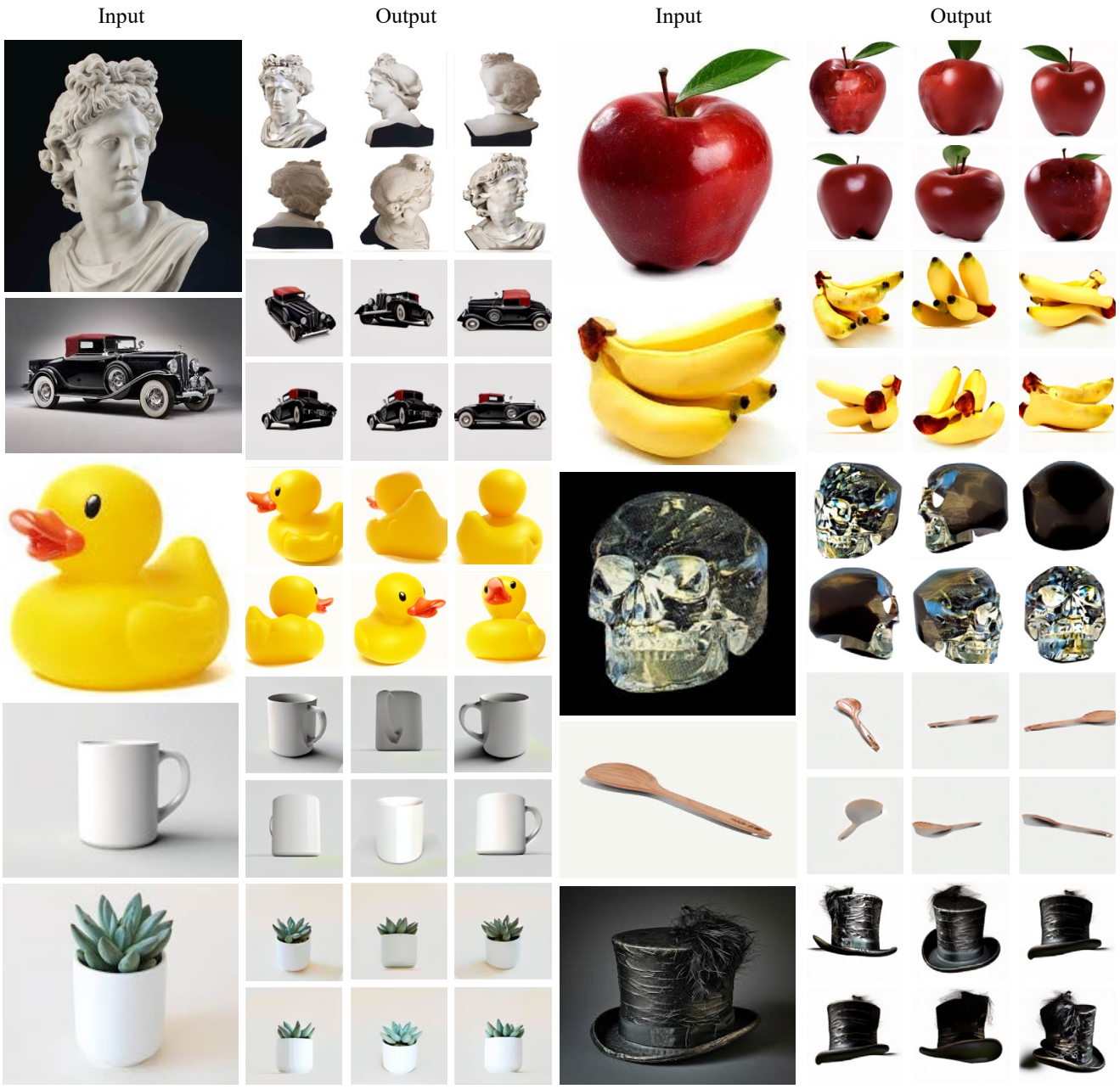


Figure 11. **Qualitative results of multi-view image generation.**

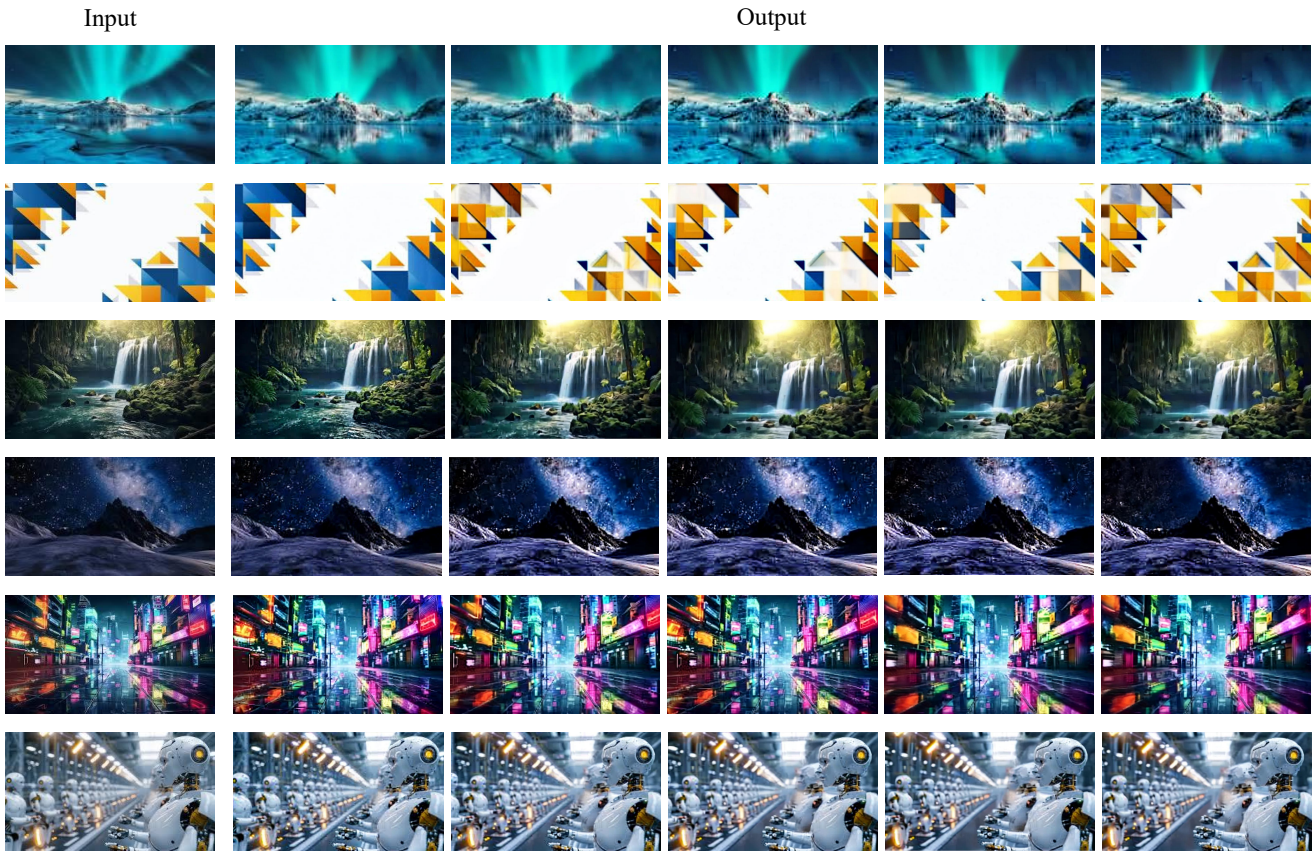


Figure 12. **Qualitative results of Image2Video.**

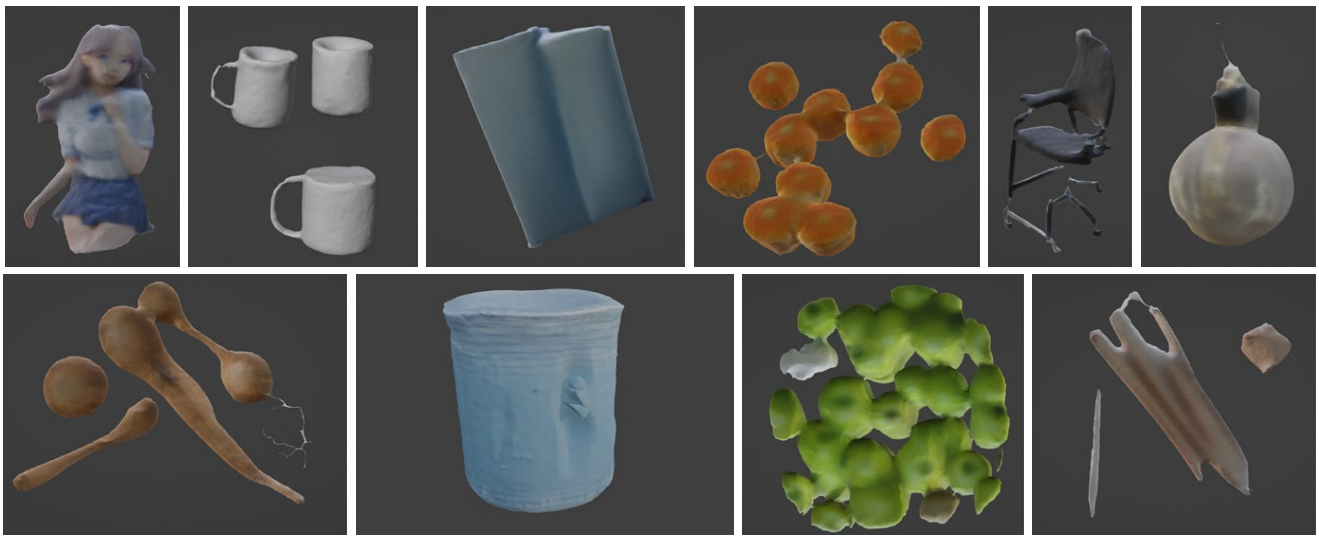


Figure 13. **Qualitative results of Text2Mesh.** The generated 3D meshes are synthesized based on their corresponding natural language instructions as specified in Table 5.