

3D Visual Prompting: Empowering 3D-Aware Visual Generation via In-Context Self-Refinement

Reina Ishikawa^{*,1,2}, Fumiaki Sato¹, Ryo Hachiuma³, Taiki Sekii^{†,1}, Hideo Saito²

¹CyberAgent, ²Keio University, ³Independent Researcher,

Correspondence: reina-ishikawa@keio.jp, taiki.sekii@gmail.com

Abstract

Recent advances in image and video generation have highlighted the need for intuitive, high-precision manipulation. However, traditional text-based prompts or 2D Visual Prompts (*e.g.*, dragging, 2D trajectories) often suffer from unintuitive prompt engineering or depth ambiguity. To address these limitations, we introduce **3D Visual Prompting**, a novel paradigm that significantly improves input intuitiveness for visual manipulation tasks. We develop a framework that integrates intuitive 3D constraints into 2D generative processes. Our approach projects 3D Visual Prompts onto 2D visual planes and leverages Multimodal Large Language Models (MLLMs) to verbalize intent, allowing plug-and-play use of existing 2D models. Furthermore, we incorporate In-Context Self-Refinement (ICSR) to summarize insights from self-correction into a meta-strategy. This enables high-precision 3D control in single-pass inference, eliminating the need for iterative computation. Experimental results demonstrate that our method significantly outperforms traditional 2D Visual Prompting for object translation and rotation. By utilizing pre-trained models without additional parameter tuning, our framework remains highly adaptable to future developments in generative models.

1 Introduction

Imagine explaining a subject’s movement in a photo to another person. Rather than relying solely on verbal descriptions, people naturally use gestures or visual cues, such as tracing trajectories or circling objects. This way of providing visual context, known as Visual Prompting, has long been recognized as an effective mode of communication. Recently, it has emerged as a powerful alternative to text prompting, allowing users to in-

* This work was done during an internship at CyberAgent.

† Corresponding author.

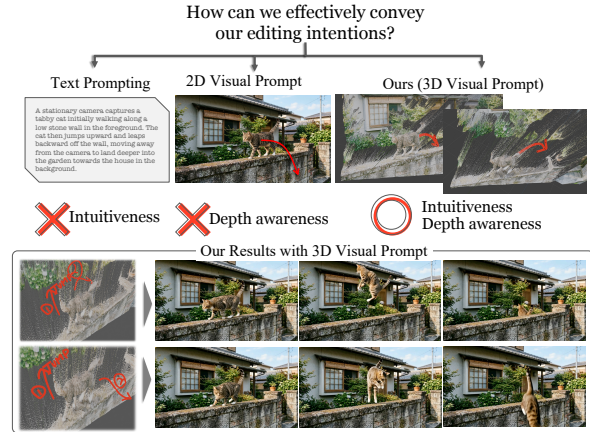


Figure 1: **3D Visual Prompting for visual manipulation tasks** (*i.e.*, image editing and video generation). Our method overcomes two key limitations of prior approaches: unintuitive prompt engineering and the depth ambiguity of 2D Visual Prompts. We enable intuitive, unambiguous control over 3D motion with 3D Visual Prompting. As illustrated below, our method correctly distinguishes whether the cat moves behind the wall or in front of it.

tuitively specify targeted visual regions (*e.g.*, pixels, bounding boxes, and motions) to Multimodal Large Language Models (MLLMs) (Yang et al., 2023a; OpenAI, 2025a; Google, 2026a; Wu et al., 2025). Such Visual Prompting has also been incorporated into generative visual manipulation tasks (*e.g.*, image editing, image-to-video generation) to intuitively specify target objects to edit (Pan et al., 2023) or draw 2D motion trajectories (Wang et al., 2024; Fang et al., 2025).

However, drawing directly on a 2D image plane (hereafter, 2D Visual Prompting) suffers from a fundamental limitation: *depth ambiguity*. For instance, a horizontal arrow over a subject may indicate motion within the image plane or along the depth axis, causing generated results to deviate from the user’s intended 3D motion. Methods that directly accept 3D inputs have also been proposed,

but many require model fine-tuning because most existing generative models are designed for text or 2D image inputs (Zheng et al., 2026; Lee et al., 2025). Even when fine-tuning is unnecessary, these methods typically assume only rigid transformations, such as translation and rotation (Pandey et al., 2024; Cheng et al., 2025), limiting their extension to more flexible motion generation, such as mid-sequence postural adjustments and spatially specific interactions with target objects. Moreover, approaches that depend on fine-tuning or access to intermediate features are often difficult to apply to closed models (e.g., GPT Image 1.5 (OpenAI, 2025b), Nano Banana Pro (Google, 2026a), and Veo (DeepMind, 2024)), and methods tied to a specific base model can quickly become obsolete as multimodal generative models evolve. This makes plug-and-play 3D-aware visual manipulation with readily available 2D generative models, without retraining, particularly important for usability and practical deployment.

To address the above issues, we make three main contributions. First, we introduce **3D Visual Prompting** to visual manipulation tasks, a new prompting paradigm that resolves the geometric ambiguities of 2D Visual Prompts and enables intuitive visual manipulation beyond simple rigid transformations. By defining prompts as visual instructions rendered in 3D space, such as text, curves, and geometric constraints, our method provides a more flexible interface for expressing 3D intent.

Second, we propose a framework that bridges the modality gap between 3D instructions and existing 2D generative models without fine-tuning. Specifically, we integrate a multimodal large language model (MLLM) into a frozen 2D generation pipeline, where a **3D-to-2D Projector** converts 3D trajectories into 2D visual representations and a **Translator** converts them into optimized textual prompts. This black-box design enables plug-and-play use of future 2D generative models.

Third, we incorporate **In-Context Self-Refinement (ICSR)** to elicit stronger 3D understanding from 2D models. An MLLM-based **Verifier** provides linguistic feedback by evaluating geometric fidelity, and iterative self-correction allows the system to accumulate meta-knowledge for communicating 3D intent. By distilling this into a universal **meta-strategy**, our framework achieves high-fidelity 3D-controlled synthesis in a single pass without test-time optimization.

Notably, our framework is designed as a *general interface layer* between 3D intent and frozen 2D generative models. Through this framework, we demonstrate that language can serve as a common interface for interpreting that intent, verifying its realization, and refining how it is communicated.

To demonstrate the efficacy of our proposed pipeline, we evaluate our method on two primary tasks: source-conditioned image editing and first-frame-conditioned video generation. We experimentally show that our approach significantly outperforms 2D Visual Prompting for complex object translation and rotation. Furthermore, extensive ablation studies quantitatively show that using distilled meta-knowledge from the ICSR process substantially enhances the system’s instruction-following capabilities.

2 Related Work

2.1 Visual Prompting

Despite the rapid advancement of MLLMs (Li et al., 2023; OpenAI et al., 2024; Bai et al., 2025), traditional text-only prompting often fails to establish precise grounding between textual descriptions and specific image regions, leading to hallucinations (Guerreiro et al., 2023). To address this, Visual Prompting (Zhang et al., 2023; Wu et al., 2024c)—which uses spatial overlays such as bounding boxes or masks—has emerged as a key paradigm for enhancing 2D visual understanding (Yang et al., 2023b). Recent research further extends this paradigm to 3D contexts, demonstrating its effectiveness in improving spatial perception (Zhang et al., 2024; Liu et al., 2025; Qi et al., 2025). The emergence of high-performance mixed-reality 3D illustration tools (Doronichev, 2016; Meta, 2026; Sosanya, 2024), along with the availability of drawing tools within simulation environments (Cohen et al., 2025), has made sketching in three-dimensional space increasingly accessible. Against this backdrop, we explicitly extend conventional 2D Visual Prompting (Wu et al., 2024a) to 3D Visual Prompting and propose a framework that facilitates intuitive three-dimensional input by leveraging advanced 3D illustration tools, enabling greater degrees of freedom in spatial interaction.

2.2 3D-Aware Visual Manipulation

The demand for precise motion control in image and video generation has shifted focus toward interaction modalities beyond text (Liu et al., 2024;

DeepMind, 2024). Following DragGAN (Pan et al., 2023), various trajectory- and point-based methods (Yin et al., 2023; Wang et al., 2024; Wu et al., 2024b; Geng et al., 2025; Fang et al., 2025) enable movement specification via 2D coordinates or direct frame annotation. While more intuitive than prompt engineering, these methods are strictly confined to the 2D plane and suffer from depth ambiguity, failing to distinguish lateral motion from movement along the depth axis.

To address these limitations, recent research incorporates 3D awareness. However, conventional approaches such as FMC (Shuai et al., 2025) and CineMaster (Wang et al., 2025b) rely on text-heavy 6D poses or bounding boxes, which are not intuitive. Other methods designed for geometric transformations (Liu et al., 2023; Cheng et al., 2025; Lee et al., 2025; Pandey et al., 2024) cannot handle complex, semantic 3D Visual Prompts. Furthermore, many 3D-aware methods require specialized architectures or fine-tuning on 3D datasets (Guo et al., 2024; Zheng et al., 2026), limiting their adaptability to rapidly evolving 2D generative models.

2.3 In-Context Self-Refinement

Prompt Adaptation (Hao et al., 2023) proposed a general framework for text-to-image generation that replaces manual prompt engineering by automatically converting user inputs into model-preferred prompts through supervised fine-tuning followed by reinforcement learning. In contrast, Prompt Diffusion (Wang et al., 2023) was among the first to equip diffusion models with In-Context Learning (ICL), enabling a single architecture to perform diverse conditional image-generation tasks, such as segmentation and depth generation, without updating model weights.

Building on ICL, In-Context Reinforcement Learning (ICRL) has been proposed as a more adaptive paradigm that improves policies through accumulated experience (Wang et al., 2025a; Moeini et al., 2025; Cai et al., 2025). Recent work, such as OmniRL (Long et al., 2025), has further shown that MLLMs can serve as effective evaluators within this framework. Inspired by this line of work, we employ ICSR, which accumulates insights through iterative self-correction without policy optimization or parameter updates, for 3D-aware visual manipulation.

To the best of our knowledge, our method is the first to introduce ICSR to 3D-aware visual manipulation, enhancing the spatial understanding of

existing 2D generative models without updating model weights.

3 Preliminaries

3.1 3D/2D Visual Prompts

We define a **3D Visual Prompt** as a set of three-dimensional editing or synthesis instructions applied to a 3D representation S_0 lifted from a source image I_0 via a mapping function $\mathcal{L}(\cdot)$ (e.g., monocular depth estimation or reference-based reconstruction), such that $S_0 = \mathcal{L}(I_0)$ can be rendered from arbitrary viewpoints. In contrast, we define a **2D Visual Prompt** as a collective term for conventional two-dimensional instructions applied directly to the source image I_0 .

3.2 Problem Formulation

To capture diverse user intents while minimizing spatial overhead, we categorize visual instructions into two distinct modalities: *3D object transformation instructions* (P^{3D}) for spatial manipulations (e.g., 3D translation, rotation, and pose), and *2D supplementary instructions* (P^{2D}) for image-level guidance (e.g., highlights and framing). Any complex editing task can be formulated as a composition of these instruction sets.

Using these prompts as conditioning variables, we address two primary tasks: (i) **Image Editing**, where given a source image I_0 , the goal is to produce an edited image $\hat{I}$ that satisfies the instructions of the Visual Prompts; and (ii) **Conditional Video Generation**, where given an initial frame I_0 , we synthesize a video sequence $\hat{V} = \{\hat{I}_t\}_{t=1}^T$ such that its temporal evolution is conditioned on the user-specified 3D transformations.

4 Methodology

4.1 Overview

Fig. 2 illustrates our proposed framework, which enables 3D-guided visual manipulation, including image editing and video generation, without requiring architectural changes or parameter fine-tuning of pre-trained 2D generative models. While most existing approaches are limited to 2D image/video inputs and natural language, our method bridges the gap between 3D spatial instructions and 2D multimodal generative models by leveraging MLLMs and ICSR. This design enables precise alignment with 3D intent while keeping model weights frozen.

We instantiate the framework with four components: (i) a **3D-to-2D Projector** ($\mathcal{R}$), (ii) a **Trans-**

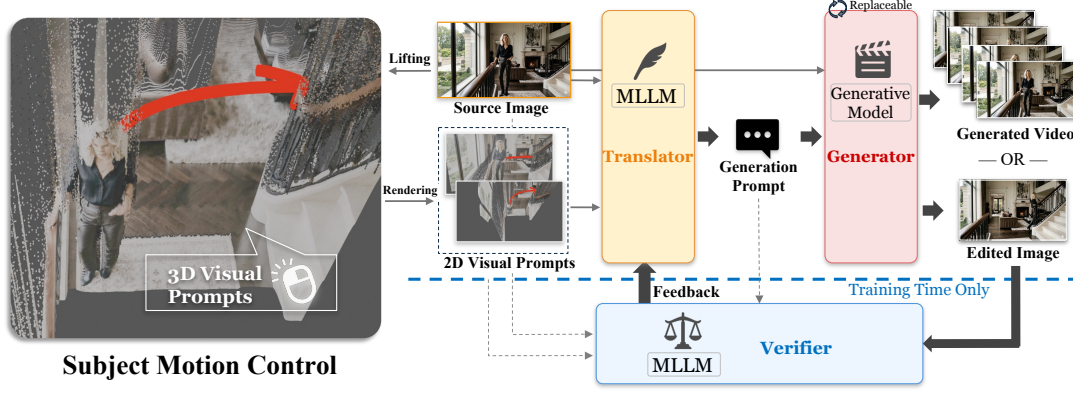


Figure 2: **Overview of our 3D Visual Prompting framework.** The framework enables 3D-guided image editing and video generation with pretrained 2D generative models, without architectural changes or fine-tuning. To bridge 3D inputs and existing 2D generative models, we project 3D Visual Prompts into 2D views and translate them into text instructions. The framework consists of two stages: an ICSR-based training phase and an inference phase using the distilled meta-strategy, enabling stronger 3D-aware generation without modifying the underlying models.

lator ($\mathcal{T}$), (iii) a **Generator** ($\mathcal{G}$), and (iv) a **Verifier** ($\mathcal{V}$). The pipeline has two stages: a **Training Phase**, where the system calibrates spatial reasoning using ground-truth outputs, and an **Inference Phase**, where it performs high-fidelity synthesis in a single pass using summarized knowledge.

It should be noted, however, that our framework is modular, and in principle, each component can be implemented with arbitrary models. Although these roles could be unified within a single model, we deliberately separate the **Translator** and the **Generator** to disentangle intent understanding from media generation. This design is motivated by the widely observed gap between discriminative and generative capabilities in existing models. In particular, feeding a single model with both multiple projected 2D views of a 3D Visual Prompt and conditioning images can easily confuse and degrade performance.

4.2 System Components

3D-to-2D Projector ($\mathcal{R}$): Since MLLMs primarily operate on 2D visual inputs, we introduce a transformation $\mathcal{R}$ that projects annotated 3D spaces into a unified 2D observable format X . We define the visual input set as:

$$X = \mathcal{R}(S_0 \oplus P^{3D}), I_0 \oplus P^{2D}. \quad (1)$$

Using the 3D scene S_0 as the underlying scene representation, $\mathcal{R}$ renders 3D Visual Prompts P^{3D} such as object trajectories. Simultaneously, the source image I_0 is augmented with 2D-specific prompts P^{2D} to provide localized semantic context. By aggregating these into X , our framework

enables joint reasoning over explicit 3D intent and the corresponding 2D semantic context.

Translator ($\mathcal{T}$): The MLLM-based Translator $\mathcal{T}$ performs two-step reasoning: intent extraction and prompt synthesis. It analyzes X to produce an explicit textual interpretation of the 3D intent t_{int} and a generative text prompt t_{gen} :

$$(t_{\text{int}}, t_{\text{gen}}) = \mathcal{T}(X). \quad (2)$$

The representation t_{int} explicitly describes 3D kinematics, ensuring spatial requirements are decoupled from the specific linguistic style of t_{gen} .

Generator ($\mathcal{G}$): Using the source frame I_0 and the synthesized prompt t_{gen} , a 2D generative model, which we refer to as a Generator $\mathcal{G}$, produces the final edited image or generated video $\hat{Y} \in \hat{I}, \hat{V}$:

$$\hat{Y} = \mathcal{G}(I_0, t_{\text{gen}}). \quad (3)$$

We note that the Generator $\mathcal{G}$ can be any visual generative model that uses text prompts as conditioning, and the framework does not require any parameter updates.

Verifier ($\mathcal{V}$): The MLLM-based Verifier $\mathcal{V}$ evaluates $\hat{Y}$ using a checklist $t_{\text{checklist}}$ generated by comparing I_0 with the ground-truth target output Y . Guided by $t_{\text{checklist}}$, it diagnoses the output media $\hat{Y}$:

$$t_{\text{fb}} = \mathcal{V}(I_0, t_{\text{int}}, t_{\text{gen}}, \hat{Y}, Y; t_{\text{checklist}}). \quad (4)$$

$\mathcal{V}$ identifies whether a discrepancy stems from a misinterpretation of 3D intent or a failure in prompt engineering, providing grounded diagnostic feedback t_{fb} .

Algorithm 1 Training via Iterative ICSR

Require: Samples $\{\mathcal{X}_i\}_{i=1}^N$, where $\mathcal{X}_i = (I_{0,i}, X_i, Y_i)$; maximum trials K
Ensure: meta-strategy $\mathcal{H}_{\text{meta}}$

- 1: $\mathcal{H}_{\text{dataset}} \leftarrow \emptyset$
- 2: **for** each sample $\mathcal{X}_i \in \{\mathcal{X}_i\}_{i=1}^N$ **do**
- 3: $\mathcal{H}_{\text{sample}}^{(0)} \leftarrow \emptyset$
- 4: $t_{\text{checklist}} \leftarrow \text{InitChecklist}(I_{0,i}, Y_i)$
- 5: **for** $k = 1$ **to** K **do**
- 6: $(t_{\text{int}}^{(k)}, t_{\text{gen}}^{(k)}) \leftarrow \mathcal{T}(X_i; \mathcal{H}_{\text{sample}}^{(k-1)})$
- 7: $\hat{Y}^{(k)} \leftarrow \mathcal{G}(I_{0,i}, t_{\text{gen}}^{(k)})$
- 8: $t_{\text{fb}}^{(k)} \leftarrow \mathcal{V}(I_{0,i}, t_{\text{int}}^{(k)}, t_{\text{gen}}^{(k)}, \hat{Y}^{(k)}, Y_i; t_{\text{checklist}})$
- 9: $\mathcal{H}_{\text{sample}}^{(k)} \leftarrow \mathcal{H}_{\text{sample}}^{(k-1)} | t_{\text{fb}}^{(k)}$
- 10: $\mathcal{H}_{\text{dataset}} \leftarrow \mathcal{H}_{\text{dataset}} | (t_{\text{gen}}^{(k)}, t_{\text{fb}}^{(k)})$
- 11: **if** $t_{\text{fb}}^{(k)}$ is success **or** $k = K$ **then**
- 12: **break**
- 13: **end if**
- 14: **end for**
- 15: **end for**
- 16: $\mathcal{H}_{\text{meta}} \leftarrow \mathcal{D}(\mathcal{H}_{\text{dataset}})$
- 17: **return** $\mathcal{H}_{\text{meta}}$

4.3 Training and Inference Phases

Our framework is designed to bridge the gap between complex 3D intent and 2D generative priors by transferring optimized strategies from a training phase to an efficient inference stage.

4.3.1 Training Phase: Iterative ICSR

The training phase aims to synthesize a universal meta-strategy $\mathcal{H}_{\text{meta}}$ by summarizing the history of self-correction processes across N training samples $\{\mathcal{X}_i\}_{i=1}^N$. Each sample is defined as a tuple $\mathcal{X}_i = (I_{0,i}, X_i, Y_i)$. As shown in Algorithm 1, the model undergoes an iterative refinement loop for each $\mathcal{X}_i$ to bridge the gap between the generated output $\hat{Y}_i$ and the ground-truth Y_i by analyzing the context in X_i .

Feedback Accumulation: For each training sample $\mathcal{X}_i$, the system executes an In-Context Self-Refinement (ICSR) loop. The Translator $\mathcal{T}$ relies on a sample-wise refinement history $\mathcal{H}_{\text{sample}}^{(k-1)} = [t_{\text{fb}}^{(1)}, \dots, t_{\text{fb}}^{(k-1)}]$ that accumulates linguistic critiques across K trials to guide the next iteration:

1. **Translation:** In iteration k , the Translator $\mathcal{T}$ analyzes the visual input X_i and the preceding refinement history to synthesize the 3D intent and generative prompt:

$$(t_{\text{int}}^{(k)}, t_{\text{gen}}^{(k)}) = \mathcal{T}(X_i; \mathcal{H}_{\text{sample}}^{(k-1)}). \quad (5)$$

2. **Verification:** The Verifier $\mathcal{V}$ assesses the generated output $\hat{Y}^{(k)}$ against the ground-truth Y_i . It produces diagnostic feedback $t_{\text{fb}}^{(k)}$, which

pinpoints specific spatial or structural discrepancies.

3. **History Update:** If the output is suboptimal, $t_{\text{fb}}^{(k)}$ is appended to the sample’s refinement history: $\mathcal{H}_{\text{sample}}^{(k)} = \mathcal{H}_{\text{sample}}^{(k-1)} | t_{\text{fb}}^{(k)}$, where $|$ denotes sequence concatenation. This updated context provides immediate refinement guidance for the subsequent trial.

History Summarization: Upon completing the iterative loops for all N samples, the aggregated refinement trajectories and feedback are collected into a dataset-wise refinement history $\mathcal{H}_{\text{dataset}}$. The History Summarizer $\mathcal{D}$ performs a meta-analysis on $\mathcal{H}_{\text{dataset}}$ to synthesize the final meta-strategy $\mathcal{H}_{\text{meta}}$:

$$\mathcal{H}_{\text{meta}} = \mathcal{D}(\mathcal{H}_{\text{dataset}}). \quad (6)$$

4.3.2 Inference Phase

During inference, the system operates in two distinct modes depending on the application of prior knowledge:

Zero-Shot Generation: In this baseline configuration, the Translator $\mathcal{T}$ interprets the 3D visual input X to synthesize a generative prompt without external guidance, as shown in Eq. (2).

Meta-Strategy-Guided Generation: To achieve superior instruction-following, the Translator’s context is augmented with the meta-strategy $\mathcal{H}_{\text{meta}}$. This acts as a strategic anchor, allowing the model to produce high-fidelity outputs by applying lessons distilled from training-time refinement without requiring iterative trials at test time:

$$(t_{\text{int}}, t_{\text{gen}}) = \mathcal{T}(X; \mathcal{H}_{\text{meta}}). \quad (7)$$

5 Experiments

To verify the effectiveness of our method, we conduct quantitative evaluations on image-editing tasks involving object translation and rotation, as well as on video-generation tasks conditioned on an initial frame depicting human motion along the depth axis. We also provide qualitative results for both image editing and video generation.

5.1 Evaluation Datasets and Metrics

Image Editing. We evaluated our framework for 3D-guided image editing on the OBJECT dataset (Michel et al., 2023), which includes 1,024

Samples	Method	Generator	Translation				Rotation			
			CLIP↑	PSNR↑	SSIM↑	LPIPS↓	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
All 1,024	Object 3DiT (Michel et al., 2023)	SD-based	—	14.400	0.262	0.498	—	16.293	0.372	0.445
	In-Video Instruction (Fang et al., 2025)	Gemini	0.920	32.720	0.701	0.274	0.901	32.259	0.698	0.266
	Ours (w/o Train)	Gemini	0.933	34.361	0.864	0.201	0.932	34.409	0.861	0.187
	Ours (w/ Train)	Gemini	0.937	34.768	0.869	0.191	0.941	35.434	0.872	0.172
	Ours (w/ Train)	GPT	0.910	33.318	0.813	0.238	0.909	32.788	0.825	0.218
Challenging 100	Ours (w/o Train)	Gemini	0.895	27.781	0.756	0.310	0.880	27.381	0.712	0.336
	Ours (w/ Train)	Gemini	0.920	30.638	0.799	0.255	0.907	29.930	0.770	0.272

SD-based: Stable Diffusion-based; Gemini: gemini-3-pro-image-preview; GPT: GPT-Image-1.5.

Table 1: Quantitative results on image editing tasks on the OBJECT dataset (Michel et al., 2023).

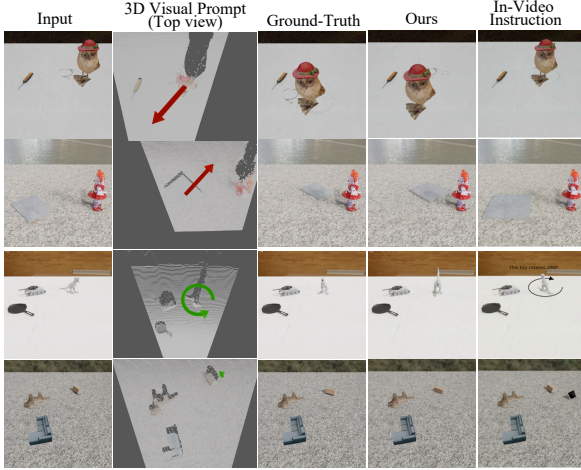


Figure 3: Image-editing results on the OBJECT dataset. Top two rows: translation; bottom two rows: rotation.

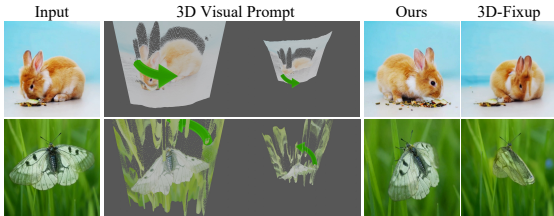


Figure 4: Qualitative comparison with the 3D-Fixup (Cheng et al., 2025).

instances each for the translation and rotation tasks. We extracted refinement histories using the proposed Iterative ICSR on the training split and evaluated performance on the unseen test split to assess generalization. We used **CLIP** (Radford et al., 2021), **PSNR**, **SSIM** (Wang et al., 2004), and **LPIPS** (Zhang et al., 2018) as evaluation metrics.

Video Generation. For video generation, we used samples from the IDEA400 subset of MotionX (Zhang et al., 2025; Lin et al., 2023). We selected 510 videos longer than five seconds with unique action labels and split them into 15 training and 495 test samples for four-second video generation. For evaluation, we uniformly sampled 60

Samples	Method	Generator	V-CLIP↑	V-PSNR↑
All 495	In-Video	—	0.893	28.598
	Ours (w/o Train)	Veo3.1	0.918	30.167
	Ours (w/ Train)	Veo3.1	0.921	30.912
	Ours (w/ Train)	Wan2.2	0.928	31.676
Challenging 100	Ours (w/o Train)	Veo3.1	0.899	26.023
	Ours (w/ Train)	Veo3.1	0.910	28.409

Veo3.1: veo-3.1-fast-generate-preview; In-Video: In-Video Instruction.

Table 2: Quantitative results on video generation tasks on the Motion-X dataset (Zhang et al., 2025; Lin et al., 2023).

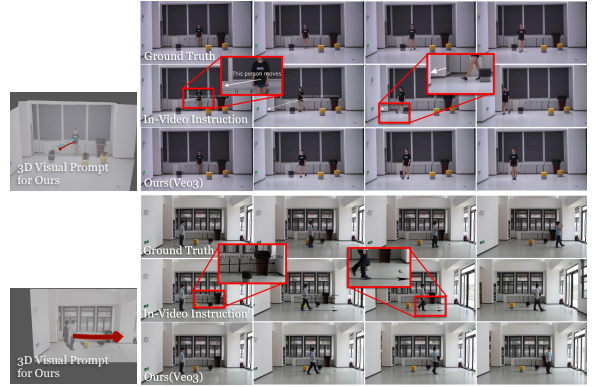


Figure 5: Video generation results on the Motion-X dataset.

frames from both the ground-truth and generated videos, computed CLIP and PSNR for each corresponding frame pair, and averaged the scores over all sampled frames. We denote these metrics as **V-CLIP** and **V-PSNR**, respectively.

5.2 Implementation Details

Since our proposed framework is modular, the Translator, Generator, and Verifier can all be utilized as plug-and-play components. Specifically, any Generator that accepts text prompts and an image as input can be integrated without fine-tuning its parameters, regardless of whether it is an open-source or closed-source model. We employed gemini-3.1-pro-preview (Google, 2026b) for both the Translator and Verifier. For the 3D-to-2D

Train	Translator	Verifier	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
✗	Gemini	–	0.895	27.781	0.756	0.310
✓	Gemini	Gemini	0.920	30.638	0.799	0.255
✓	Gemini	GPT	0.938	30.678	0.816	0.235
✗	GPT	–	0.931	30.653	0.818	0.237
✓	GPT	GPT	0.934	30.858	0.819	0.232
✓	GPT	Gemini	0.932	30.506	0.817	0.236
✗	Qwen	–	0.915	28.917	0.792	0.279
✓	Qwen	Qwen	0.918	29.755	0.812	0.262

Gemini: gemini-3.1-pro-preview; GPT: GPT-5.1; Qwen: Qwen3-VL-8B-Instruct.

Table 3: Ablation on Translator and Verifier models.

Train	Summarizer	# Tokens	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
✗	–	–	0.895	27.781	0.756	0.310
✓	Gemini	661	0.927	29.641	0.803	0.245
✓	GPT	1,127	0.920	30.638	0.799	0.255
✓	None (Direct)	8,825	0.929	30.356	0.813	0.239

Gemini: gemini-3.1-pro-preview; GPT: GPT-5.1.

Table 4: Ablation on History Summarizer model. Tokens denote the length of the resulting meta-strategy, measured using c1100k_base (OpenAI, 2022).

Projector, we first lifted the source image into a 3D representation using MoGe-2 (Wang et al., 2025c). Within this reconstructed space, we annotated 3D motions for target subjects using Blender (Cohen et al., 2025).

Unless otherwise specified, we used two rendered images for the 3D Visual Prompt (*i.e.*, the front and top views) and adopted the image labeled *Final centroid position* as the 2D Visual Prompt. For the *w/ Train* setting, we used History Summarization based on 15 training samples and 5 ICSR iterations, where the best and worst iterations of each sample were selected based on a combined metric score ($CLIP + PSNR/50 + SSIM - LPIPS$) and summarized by GPT-5.1 (OpenAI, 2025a). We resized all output images or frames to the ground-truth resolution using bilinear interpolation and computed the metrics at the unified resolution.

5.3 Results on Image Editing Tasks

Tab. 1 reports the results using gemini-3-pro-image-preview and GPT-Image-1.5 (OpenAI, 2025b). Our method substantially outperforms Object 3DiT and In-Video Instruction, demonstrating strong 3D-aware editing performance without tuning the underlying generative model. Notably, our zero-shot setting (*w/o Train*) already surpasses the baselines, while meta-strategy-guided generation (*w/ Train*) further improves performance.

The improvement is more pronounced on challenging samples: among the 100 lowest-performing samples under *w/o Train*, selected

Train	History Sel.	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
✗	–	0.895	27.781	0.756	0.310
✓	Best-Worst	0.920	30.638	0.799	0.255
✓	Random	0.932	29.979	0.803	0.246
✓	All	0.925	29.801	0.802	0.245

Table 5: Ablation on training history selection, comparing three strategies for selecting prompt–feedback pairs.

Train	Sample Sel.	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
✗	–	0.895	27.781	0.756	0.310
✓	Random	0.920	30.638	0.799	0.255
✓	Diversity	0.928	29.643	0.804	0.248
✓	Nearest-Neighbor	0.932	29.908	0.806	0.246
✓	Metric	0.927	29.940	0.806	0.244

Table 6: Ablation on training sample selection. Each strategy selects 15 of 100 candidates. *Random* randomly selects samples, *Diversity* selects samples with diverse similarity to the test input, *Nearest-Neighbor* selects the most similar samples, and *Metric* selects samples with the highest combined metric scores.

based on the combined metric score ($CLIP + PSNR/50 + SSIM - LPIPS$), the performance gap between *w/o* and *w/ Train* increases substantially. We provide further analysis with varying numbers of challenging samples in Appendix B.

We attribute the advantage over In-Video Instruction to two main factors. First, directly overlaying instructions on the conditioning image can degrade generation quality, as the instructions may remain in the output. Second, relying solely on 2D visual cues is susceptible to 3D ambiguity, particularly in translation, leading to incorrect motion directions (Fig. 3). Fig. 4 further provides a qualitative comparison with 3D-Fixup (Cheng et al., 2025). Since its code is not publicly available, we restrict the comparison to qualitative results. Overall, despite not fine-tuning the underlying generative model, our method achieves 3D-aware editing performance qualitatively comparable to that of 3D-Fixup, which is fine-tuned for this task, while preserving source image quality.

5.4 Results on the Video Generation Task

Tab. 2 reports the results of our method using either veo-3.1-fast-generate-preview (DeepMind, 2024) (hereafter Veo3.1) or Wan2.2 (Wan et al., 2025), together with a quantitative comparison against In-Video Instruction, a baseline designed for controllable video generation. On Veo3.1, our method substantially outperforms the baseline even in the *w/o Train* setting, and performance further improves slightly in the *w/ Train*

Train	# Samples	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
$\times$	—	0.895	27.781	0.756	0.310
✓	5	0.926	29.546	0.802	0.250
✓	15	0.920	30.638	0.799	0.255
✓	30	0.926	29.780	0.801	0.253

Table 7: Ablation on the number of training samples.

Train	# Iterations	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
$\times$	—	0.895	27.781	0.756	0.310
✓	5	0.920	30.638	0.799	0.255
✓	10	0.933	30.217	0.808	0.237

Table 8: Ablation on the number of ICSR iterations.

setting. We additionally report results with Wan2.2, where the proposed framework yields strong performance, suggesting compatibility with different video generation models. The performance improvement from *w/o Train* to *w/ Train* becomes more pronounced for the worst-performing samples in the *w/o Train* setting, consistent with the trend observed in the image editing results.

The reason for the performance gap between the baseline and our method becomes clear from the qualitative results in Fig. 5. In In-Video Instruction, the 2D Visual Prompts written directly on the initial frame are often interpreted as part of the scene and remain visible, leading to unintended motion. Moreover, because the method relies solely on 2D guidance, it is more susceptible to depth ambiguity and may misinterpret occlusion relationships, resulting in incorrect motion paths. In contrast, our method captures the front-back relationships among surrounding objects, leading to more accurate motion generation.

5.5 Ablation Study

We comprehensively evaluate key components and design choices of our framework on the image-editing translation task using the 100 challenging samples defined above.

MLLM Selection. We evaluate different MLLMs for the Translator and Verifier, including gemini-3.1-pro-preview, GPT-5.1, and Qwen3-VL-8B-Instruct (Bai et al., 2025), as well as cross-model configurations. As shown in Tab. 3, *w/ Train* settings consistently outperform their corresponding *w/o Train* settings across different MLLMs, demonstrating that our framework is not tied to a specific MLLM and supports plug-and-play model replacement. Meanwhile, the observed performance differences across MLLMs suggest that our framework may further benefit

# Rendered Images	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
2 (front & top)	0.895	27.781	0.756	0.310
3 (front & top & side)	0.930	29.786	0.795	0.257

Table 9: Ablation on the number of rendered images.

Factor	Compared Settings	Preferred	Preference
Visual Prompt Training	3D vs. 2D <i>w/</i> vs. <i>w/o Train</i>	3D <i>w/ Train</i>	78% 71%

Table 10: Pairwise preference evaluation using GPT-5.1. 3D vs. 2D is evaluated *w/o Train*, while *w/* vs. *w/o Train* is evaluated with 3D Visual Prompts.

from future advances in MLLM capabilities. The results in Tab. 4 show comparable performance between Gemini and GPT, indicating limited sensitivity to the summarizer choice, while both substantially reduce context length compared with direct history input.

Training History Selection. We compare strategies for selecting prompt-feedback pairs from the training histories. As shown in Tab. 5, all strategies achieve comparable performance, suggesting that selecting a compact subset can maintain performance while potentially reducing the input cost associated with redundant histories.

Training Sample Selection. We further examine the selection of training samples from the candidate set. As shown in Tab. 6, all strategies substantially outperform the *w/o Train* setting, with relatively small differences among them, suggesting that the performance improvement is not strongly dependent on the sample selection strategy.

Training Configuration. As shown in Tab. 7, performance improves with only five training samples, demonstrating the effectiveness of our framework with limited training data. Tab. 8 further shows that increasing the number of iterations from 5 to 10 tends to improve performance overall, although the differences remain relatively small, suggesting that five iterations are sufficient to achieve most of the improvement.

3D Visual Prompt Design. Tab. 9 shows that increasing the number of rendered viewpoints from two to three tends to improve performance. This suggests that additional viewpoints can help interpret 3D spatial information.

Spatial Expressiveness. We conduct a pairwise preference evaluation using GPT-5.1 to examine whether 3D Visual Prompts and ICSR improve the



Figure 6: **Advanced application examples.**

Train Perturbation	CLIP↑	PSNR↑	SSIM↑	LPIPS↓
✗ None (Default)	0.895	27.781	0.756	0.310
✓ None (Default)	0.920	30.638	0.799	0.255
✓ Quantization (Lv. 100)	0.921	30.083	0.796	0.259
✓ Quantization (Lv. 10)	0.927	30.446	0.808	0.243
✓ Noise ([0, 50))	0.929	30.321	0.806	0.245
✓ Noise ([0, 150))	0.925	30.115	0.801	0.252

Table 11: Robustness to perturbations in 3D Visual Prompts. *Quantization* discretizes the depth dimension into 100 or 10 levels. *Noise* adds uniform noise within the specified range to each rendered image pixel.

Input method	Time (s)	V-CLIP↑	V-PSNR↑
Text Prompting	143.4	-	-
2D Visual Prompting	21.9	0.876	29.537
3D Visual Prompting	25.0	0.923	32.129

Table 12: Usability analysis on input methods.

spatial expressiveness of the prompts generated by the Translator. As reported in Tab. 10, 3D Visual Prompts are preferred over 2D Visual Prompts in 78% of comparisons, while *w/ Train* is preferred over *w/o Train* in 71%. These results support our framework as a *general interface layer* for conveying 3D intent to frozen 2D generative models.

5.6 Robustness Analysis

We evaluate robustness to perturbations in 3D Visual Prompts on the same 100 challenging samples (Tab. 11). Performance remains comparable to the unperturbed *w/ Train* setting across different perturbations, with all perturbed settings consistently outperforming *w/o Train* across all metrics. These results indicate that the MLLM can effectively interpret 3D Visual Prompts even with noisy depth, and that highly accurate 3D reconstruction is not required. This robustness may also stem from our framework not directly feeding noisy 3D Visual Prompts into the generative models.

5.7 Usability Analysis

We conducted a usability study with four non-author participants, each using three input methods on five video-generation tasks: Text Prompting, GUI-based 2D Visual Prompting, and GUI-based 3D Visual Prompting. Tab. 12 indicates that text prompting requires substantially more input time than both Visual Prompting methods, while 3D Visual Prompting achieves higher fidelity than 2D Visual Prompting with a small input-time overhead.

5.8 Advanced Applications

In the preceding experiments, we restricted the evaluation to simple geometric transformations to enable quantitative comparison. However, a key strength of our method lies in its ability to support more flexible forms of manipulation. To demonstrate this, we further applied our approach to video generation using more advanced hand-drawn 3D Visual Prompts. Fig. 6 shows representative results. These examples suggest that our method can accommodate expressive free-form user inputs.

6 Conclusion

In this paper, we introduced **3D Visual Prompting**, a model-agnostic framework for 3D-aware visual manipulation. By integrating intuitive 3D geometric constraints into existing 2D generative models, our method enables precise 3D-aware control over image and video generation. Moreover, incorporating In-Context Self-Refinement (ICSR) enables more accurate single-pass 3D control without modifying the underlying architecture. Extensive experiments demonstrate consistent improvements over conventional 2D Visual Prompting across image and video tasks, as well as the flexibility and robustness of our framework across different models and design choices.

Limitations

Our framework is fully modular: the Generator, Translator, and Verifier can all be replaced without fine-tuning. However, this flexibility also means that the overall system performance depends heavily on the capabilities of the selected models. Therefore, to achieve high-precision visual manipulation, it is necessary to select models that are well matched to the target task. As our experiments suggest, this does not require models explicitly trained to be 3D-aware, but it does require sufficiently strong visual understanding and generation capabilities.

This limitation is particularly clear when the component models are mismatched to the task. For example, a Generator specialized for style transfer is unlikely to handle motion-related edits, such as object translation, effectively. Likewise, if the Translator or Verifier fails to correctly interpret Visual Prompts—for instance, by misreading the direction of an arrow—the system may produce incorrect outputs. In practice, reliable performance depends on strong generative models, such as those adopted in this work.

However, this dependency also highlights an important advantage of our framework. Because it does not rely on task-specific fine-tuning, it can directly benefit from advances in general-purpose MLLMs and generative models. As these models continue to improve and become more accessible, the overall system can improve accordingly without any modification to the framework itself.

Ethical Considerations

Our framework is designed to enable flexible visual manipulation and is not intended to cause direct negative social impacts. Nevertheless, its use is ultimately determined by the user, and its outputs depend on the behavior of the underlying models incorporated into the system. As a result, harmful or offensive images and videos may still be generated under certain inputs or model configurations.

Accordingly, users should be aware of these potential risks and use the system responsibly. In particular, the framework should not be deliberately used to generate abusive, harmful, or otherwise inappropriate content targeting specific individuals.

Acknowledgment

This work was supported by JST BOOST, Japan Grant Number JPMJBS2409.

References

- Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, and Chunjiang Ge et al. 2025. [Qwen3-vl technical report](#). *Preprint*, arXiv:2511.21631.
- Zhicheng Cai, Xinyuan Guo, Yu Pei, Jiangtao Feng, Jinsong Su, Jiangjie Chen, Ya-Qin Zhang, Wei-Ying Ma, Mingxuan Wang, and Hao Zhou. 2025. [Flex: Continuous agent evolution via forward learning from experience](#). *Preprint*, arXiv:2511.06449.
- Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, and Andrew Huang et al. 2025. [Sam 3: Segment anything with concepts](#). *Preprint*, arXiv:2511.16719.
- Yen-Chi Cheng, Krishna Kumar Singh, Jae Shin Yoon, Alexander Schwing, Liang-Yan Gui, Matheus Gadelha, Paul Guerrero, and Nanxuan Zhao. 2025. [3d-fixup: Advancing photo editing with 3d priors](#). In *SIGGRAPH*. ACM.
- Fiona Cohen, Francesco Siddi, Dalai Felinto, and Ton Roosendaal. 2025. [Blender foundation annual report 2024](#). Technical report.
- Google DeepMind. 2024. [Veo: A text-to-video generation system](#). Technical report.
- Andrey Doronichev. 2016. [Tilt brush: painting from a new perspective](#). Accessed: 2026-08-30.
- Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. [In-video instructions: Visual signals as generative control](#). *Preprint*, arXiv:2511.19401.
- Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, Oliver Wang, Andrew Owens, and Deqing Sun. 2025. Motion prompting: Controlling video generation with motion trajectories. In *CVPR*.
- Google. 2026a. [Gemini 3 pro image \(nano banana pro\)](#). Accessed: 2026-08-30.
- Google. 2026b. [Gemini 3.1 pro preview](#). Accessed: 2026-08-30.
- Nuno M. Guerreiro, Elena Voita, and André Martins. 2023. [Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation](#). In *EACL*.

- Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. 2024. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. In *ICLR*.
- Yaru Hao, Zewen Chi, Li Dong, and Furu Wei. 2023. Optimizing prompts for text-to-image generation. In *NeurIPS*.
- Yao-Chih Lee, Zhoutong Zhang, Jiahui Huang, Jui-Hsien Wang, Joon-Young Lee, Jia-Bin Huang, Eli Shechtman, and Zhengqi Li. 2025. [Generative video motion editing with 3d point tracks](#). *Preprint*, arXiv:2512.02015.
- Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. Blip-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*.
- Jing Lin, Ailing Zeng, Shunlin Lu, Yuanhao Cai, Ruimao Zhang, Haoqian Wang, and Lei Zhang. 2023. Motion-x: A large-scale 3d expressive whole-body human motion dataset. *NeurIPS*.
- Dingning Liu, Cheng Wang, Peng Gao, Renrui Zhang, Xinzhu Ma, Yuan Meng, and Zhihui Wang. 2025. [3daxisprompt: Promoting the 3d grounding and reasoning in gpt-4o](#). *Neurocomput.*, 637(C).
- Ruoshi Liu, Rundui Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. 2023. [Zero-1-to-3: Zero-shot one image to 3d object](#). In *ICCV*.
- Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, Lifang He, and Lichao Sun. 2024. [Sora: A review on background, technology, limitations, and opportunities of large vision models](#). *Preprint*, arXiv:2402.17177.
- Rujiao Long, Ziyu Jin, Zhan Wang, Zijin Huang, Qian-nan Cheng, and Lei Yi. 2025. [Omnirl: Omni reward and loss scheme for vision-language rl model training](#). Accessed: 2025-03-18.
- Meta. 2026. [Introducing ‘shapexr,’ a new tool for real-time collaboration and prototyping in vr](#). Accessed: 2026-08-30.
- Oscar Michel, Anand Bhattad, Eli VanderBilt, Ranjay Krishna, Aniruddha Kembhavi, and Tanmay Gupta. 2023. Object 3dit: language-guided 3d-aware image editing. *NeurIPS*.
- Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangdong Zhang. 2025. [A survey of in-context reinforcement learning](#). *Preprint*, arXiv:2502.07978.
- OpenAI. 2022. [tiktoken](#). Accessed: 2026-08-30.
- OpenAI. 2025a. [Gpt-5.1: A smarter, more conversational chatgpt](#). Accessed: 2026-08-30.
- OpenAI. 2025b. [The new chatgpt images is here](#). Accessed: 2026-08-30.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, and Sam Altman et al. 2024. [Gpt-4 technical report](#). *Preprint*, arXiv:2303.08774.
- Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. [Drag your gan: Interactive point-based manipulation on the generative image manifold](#). In *SIGGRAPH*. ACM.
- Karran Pandey, Paul Guerrero, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, and Niloy J. Mitra. 2024. Diffusion handles enabling 3d edits for diffusion models by lifting activations to 3d. In *CVPR*.
- Zhangyang Qi, Zhixiong Zhang, Ye Fang, Jiaqi Wang, and Hengshuang Zhao. 2025. [Gpt4scene: Understand 3d scenes from videos with vision-language models](#). *Preprint*, arXiv:2501.01428.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML*.
- Xincheng Shuai, Henghui Ding, Zhenyuan Qin, Hao Luo, Xingjun Ma, and Dacheng Tao. 2025. Free-form motion control: Controlling the 6d poses of camera and objects in video generation. In *ICCV*.
- Oluwaseyi Sosanya. 2024. [Redefining design: Gravity sketch and the evolution of creative tools](#).
- Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, and Jianxiao Yang et al. 2025. [Wan: Open and advanced large-scale video generative models](#). *Preprint*, arXiv:2503.20314.
- Fan Wang, Pengtao Shao, Yiming Zhang, Bo Yu, Shaoshan Liu, Ning Ding, Yang Cao, Yu Kang, and Haifeng Wang. 2025a. [Towards large-scale in-context reinforcement learning by meta-training in randomized worlds](#). *Preprint*, arXiv:2502.02869.
- Qinghe Wang, Yawen Luo, Xiaoyu Shi, Xu Jia, Huchuan Lu, Tianfan Xue, Xintao Wang, Pengfei Wan, Di Zhang, and Kun Gai. 2025b. [Cinemaster: A 3d-aware and controllable framework for cinematic text-to-video generation](#). In *SIGGRAPH*. ACM.
- Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. 2025c. [Moge-2: Accurate monocular geometry with metric scale and sharp details](#). *Preprint*, arXiv:2507.02546.

- Zhendong Wang, Yifan Jiang, Yadong Lu, Yelong Shen, Pengcheng He, Weizhu Chen, Zhangyang Wang, and Mingyuan Zhou. 2023. In-context learning unlocked for diffusion models. In *NeurIPS*.
- Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. 2004. [Image quality assessment: from error visibility to structural similarity](#). *IEEE Trans.*, 13(4):600–612.
- Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. 2024. [Motionctrl: A unified and flexible motion controller for video generation](#). In *SIGGRAPH*. ACM.
- Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, and Yilei Chen et al. 2025. [Qwen-image technical report](#). *Preprint*, arXiv:2508.02324.
- Junda Wu, Zhehao Zhang, Yu Xia, Xintong Li, Zhaoyang Xia, Aaron Chang, Tong Yu, Sungchul Kim, Ryan A. Rossi, Ruiyi Zhang, Subrata Mitra, Dimitris N. Metaxas, Lina Yao, Jingbo Shang, and Julian McAuley. 2024a. [Visual prompting in multi-modal large language models: A survey](#). *Preprint*, arXiv:2409.15310.
- Weijia Wu, Zhuang Li, Yuchao Gu, Rui Zhao, Yefei He, David Junhao Zhang, Mike Zheng Shou, Yan Li, Tingting Gao, and Di Zhang. 2024b. [Draganything: Motion control for anything using entity representation](#). In *ECCV*.
- Yixuan Wu, Yizhou Wang, Shixiang Tang, Wenhao Wu, Tong He, Wanli Ouyang, Philip Torr, and Jian Wu. 2024c. [Dettoolchain: A new prompting paradigm to unleash detection ability of mllm](#). In *ECCV*.
- Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023a. [Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v](#). *Preprint*, arXiv:2310.11441.
- Lingfeng Yang, Yueze Wang, Xiang Li, Xinlong Wang, and Jian Yang. 2023b. Fine-grained visual prompting. In *NeurIPS*.
- Shengming Yin, Chenfei Wu, Jian Liang, Jie Shi, Houqiang Li, Gong Ming, and Nan Duan. 2023. [Dragnuwa: Fine-grained control in video generation by integrating text, image, and trajectory](#). *Preprint*, arXiv:2308.08089.
- Ao Zhang, Hao Fei, Yuan Yao, Wei Ji, Li Li, Zhiyuan Liu, and Tat-Seng Chua. 2023. Vpgrans: Transfer visual prompt generator across llms. In *NeurIPS*.
- Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Sha Zhang, Di Huang, Jiajun Deng, Shixiang Tang, Wanli Ouyang, Tong He, and Yanyong Zhang. 2024. [Agent3d-zero: An agent for zero-shot 3d understanding](#). In *ECCV*.
- Yuhong Zhang, Jing Lin, Ailing Zeng, Guanlin Wu, Shunlin Lu, Yurong Fu, Yuanhao Cai, Ruimao Zhang, Haoqian Wang, and Lei Zhang. 2025. Motion-x++: A large-scale multimodal 3d whole-body human motion dataset. *arXiv preprint arXiv:2501.05098*.
- Sixiao Zheng, Minghao Yin, Wenbo Hu, Xiaoyu Li, Ying Shan, and Yanwei Fu. 2026. Versecrafter: Dynamic realistic video world model with 4d geometric control. *arXiv preprint arXiv:2601.05138*.

A Implementation Details of the 3D-to-2D Projector

2D-to-3D Lifting. For lifting the source image into a point cloud, we use MoGe (Wang et al., 2025c) and perform floor alignment based on percentile statistics. This floor alignment ensures that, regardless of the input capture conditions, the rendered views, such as the front and top views, are geometrically consistent; specifically, the angle between the ground plane and the optical axis of the rendering camera is kept constant.

Camera Look-at Point in Blender. The camera look-at point is determined according to the availability of a clearly identifiable target object. When the target category can be explicitly specified, we apply segmentation with SAM3 (Carion et al., 2025) and use the centroid of the segmented point cloud as the look-at point. Since the Motion-X dataset targets only humans, we adopt this strategy for that dataset. In contrast, for the OBJECT dataset, we use the centroid of the entire point cloud as the look-at point.

3D-to-2D Rendering Camera Transformation. We define the front, top, and side views using two elliptical camera orbits centered on the target object in Blender coordinates: a *vertical orbit* on the XZ plane and a *horizontal orbit* lying on a plane parallel to the XY plane. To improve the visibility of the 3D Visual Prompt, the front view is placed on the horizontal orbit with an elevation offset of 30% along the vertical direction (for Motion-X, we use 10% instead). The top view is taken from a major-axis vertex of the horizontal orbit. The side view is taken from a major-axis vertex of the vertical orbit that passes through the front-view position. For each dataset, the axis lengths of the elliptical orbits are determined empirically based on the distance between the target object and the source camera, such that the rendered view covers approximately $2\times$ to $4\times$ that distance. When a larger number of rendered views is required, we render images by moving the camera at uniform intervals along these orbits. All 2D renderings are performed at a fixed resolution of 1024×768 .

B Analysis on Challenging Samples

We further analyze the effect of ICSR on challenging samples by varying the size of the worst-performing subset. Specifically, we rank samples based on the combined metric score ($CLIP +$

$PSNR/50 + SSIM - LPIPS$) under the *w/o Train* setting and evaluate *w/ Train* and *w/o Train* on the worst- K samples for different values of K .

Figures 7a and 7b demonstrate that the performance gap between *w/ Train* and *w/o Train* is consistently larger for smaller K across the image-editing translation and rotation tasks. As K increases to include easier samples, the gap gradually decreases across the evaluation metrics. A similar trend is observed for video generation in Fig. 8, where the improvement from training is particularly pronounced for smaller worst- K subsets.

These results indicate that ICSR is particularly effective on challenging samples where the zero-shot setting struggles, while maintaining improvements as the evaluation expands to a broader set of samples.

C Details of Usability Analysis

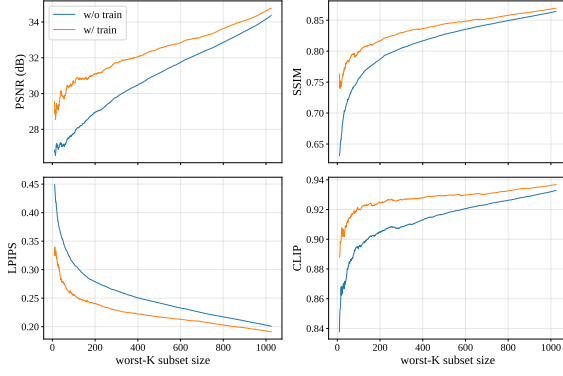
C.1 Participants and Consent

Four participants were voluntarily recruited through personal contacts without compensation. Before participation, all participants received a written description of the study and provided written informed consent. Participants were informed that they could withdraw their consent at any time without disadvantage.

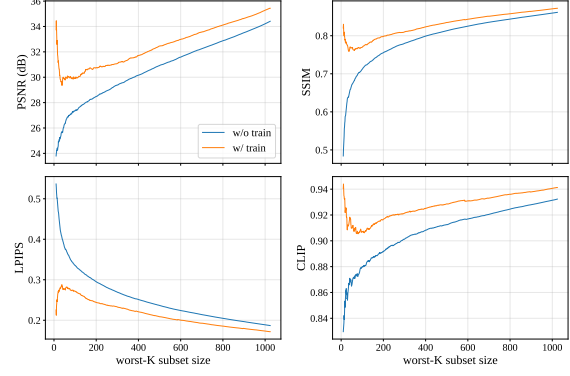
We collected only the inputs produced during the study (i.e., text prompts or drawn trajectories) and the time required to complete each input. Participants were informed that the collected data would be used only for research purposes and that results reported in academic publications would be anonymized or aggregated.

C.2 Instructions and Interface

Instructions Given to Participants. We provided the following task instructions to all participants before the study. The original instructions were provided in Japanese; we report an English translation below.



(a) Image-editing translation task.



(b) Image-editing rotation task.

Figure 7: **Effect of ICSR on challenging samples for image editing.** Performance on the worst- K samples ranked under the *w/o Train* setting. Smaller K corresponds to more challenging samples.

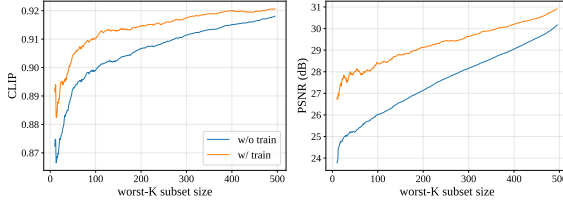


Figure 8: **Effect of ICSR on challenging samples for video generation.** Performance on the worst- K samples ranked under the *w/o Train* setting. Smaller K corresponds to more challenging samples.

In this study, you will watch a four-second video of a person moving and create an input for generating a video with the same motion.

You will use each of the following three input methods for five videos:

- Text Prompt: Enter an English text prompt describing the motion.
- 2D Visual Prompt: Draw an arrow on a two-dimensional image.
- 3D Visual Prompt: Draw an arrow in three-dimensional space.

Before the experiment, you will complete several practice trials. Once the experimenter starts the application on the PC, please begin creating the input.

This experiment is not intended to assess your individual abilities. If you experience any physical or psychological discomfort during the experiment, please inform the experimenter. You may stop the experiment at any time.

Interface and Generation Procedure. Fig. 9 shows the interfaces used for 2D and 3D Visual Prompting.

Text Prompting. Participants watched the reference video and typed an English prompt to describe its motion. The text prompt was directly used as input for video generation.

2D Visual Prompting. Participants watched the reference video and drew an arrow directly on its first frame, as shown in Fig. 9a. For video generation, the arrow was overlaid on the first frame, and the resulting image was directly used as input.

3D Visual Prompting. Participants watched the reference video and drew an arrow on a top-view projection obtained by lifting the first frame into 3D. Since the Motion-X samples used in our study primarily involved planar motion, the top view served as a suitable default for specifying motion. As shown in Fig. 9b, we also provided an interactive 3D visualization tool that allowed participants to inspect the lifted scene and projected arrow. The specified 3D Bézier curve was then re-rendered in Blender and used as a 3D Visual Prompt. The resulting prompt was processed by our framework with the *w/ Train* setting to generate the video.

D Details of Pairwise Preference Evaluation

For the pairwise preference evaluation, we use GPT-5.1 with high reasoning effort to compare the final editing prompts produced by two settings. To avoid bias toward a particular method, the two prompts are presented anonymously as *Prompt A* and *Prompt B*. The evaluator is instructed to select the prompt that expresses the intended motion more three-dimensionally based solely on the prompt text. We apply this evaluation to compare

3D and 2D Visual Prompts and to compare *w/ Train* and *w/o Train*. The evaluator returns the preferred prompt together with a one-sentence justification in JSON format. The evaluation prompt is shown below:

```
You compare two short English editing
prompts for an image model.
Decide which prompt expresses motion more
three-dimensionally.

Do not assume either label corresponds
to any particular pipeline. Base your
judgment only on the text.

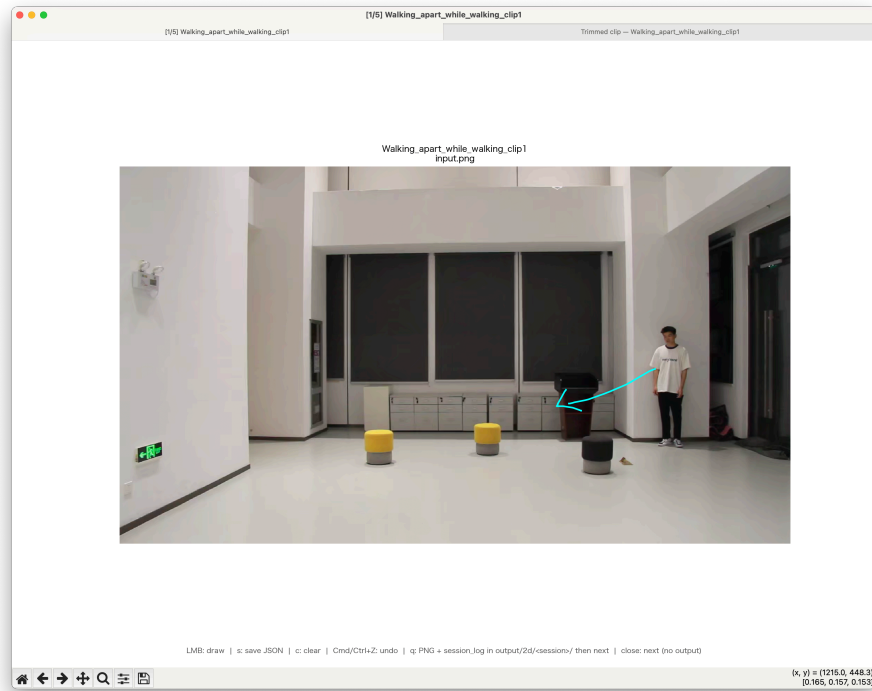
Prompt A:
-
{Prompt A}
-

Prompt B:
-
{Prompt B}
-

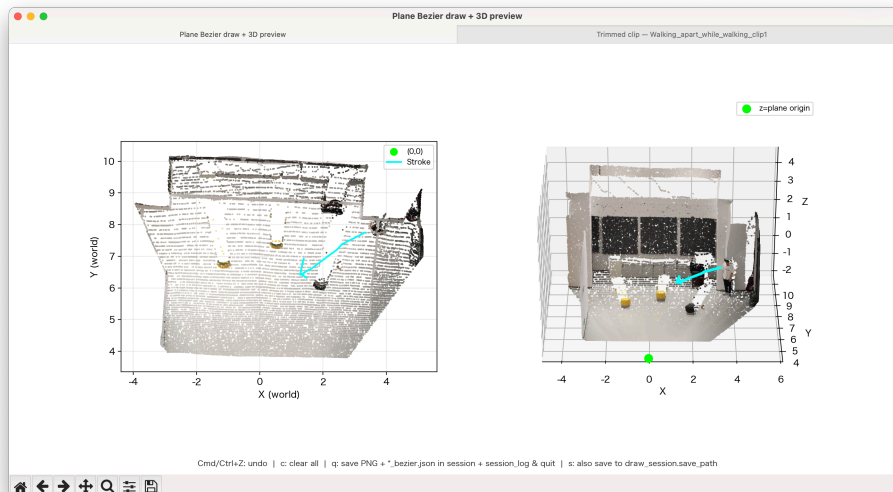
Reply with a single JSON object and
nothing else:
{"winner": "A" or "B", "reason": "one short
sentence"}
```

E Prompt Templates

Figures 10–23 present the prompts provided to the Translator, Verifier, and History Summarizer.



(a) Example interface for 2D Visual Prompting.



(b) Example interface for 3D Visual Prompting.

Figure 9: **Interfaces used in the usability study.** In 2D Visual Prompting, participants draw an arrow directly on the first video frame. In 3D Visual Prompting, participants draw an arrow on the top-view projection of the lifted scene while inspecting its 3D visualization.

```
# Role:
You are a helpful assistant for 3D-aware {image editing/video generation} tasks.
Your goal is to generate an optimized prompt for the intended 3D-aware {image editing/video generation}.
To achieve this, first you need to understand the editing intention and the source image{ (the first frame of the video)}, then generate an optimized prompt.
```

Figure 10: System prompt template for the Translator

```
# Step1 [Source Image Analysis]:
You are given a source image before editing. Analyze the source image following the instructions below.

## Visual Contexts:
- Source Image (The image before the motion. We define the camera that captured this image as the "Reference Camera.")

## Task:
1. Describe the location of the category(s) and orientation in the source image in detail. Be careful not to become ambiguous depending on the viewpoint.
2. Please interpret the meaning of any visual cues or text added to the image.
```

Figure 11: User Prompt template for the Translator: Step 1

```
# Step2 [Motion Description]:
The {category}(s) begin moving or rotating from the position shown in the Source Image. Analyze the motion following the instructions below.

## Visual Contexts:
- **Point Cloud:** The point cloud from the Source Image viewed from different perspectives. {description of visual prompts}
- **Camera Frustum Marker:** Indicates the Reference Camera's position and orientation for the Source Image.

## Expected Motion Indicators:
Interpret the movement by identifying which of the following indicators are present (one or more may appear). Identify by Color.
- **Red** Arrow: The arrow head indicates the {category}(s)'s expected motion trajectory.
- **Cyan** Triangle: The apex of the triangle indicates the {category}(s)'s expected orientation change at specific points along the trajectory.
- **Green** Circular Arrow: Indicates that the {category}(s) spins on the spot. The arrow's direction and angle define the spin's rotation.

## Task:
Describe the {category}(s)'s 3D movement in detail while also taking into account the interpreted meaning of the visual cues provided in the Source Image. Focus specifically on the depth-wise final position/orientation relative to the initial state.
Be careful not to become ambiguous from different viewpoints. Use forward to describe the direction closer to the camera, and backward to describe the direction farther from the camera.
```

Figure 12: User prompt template for the Translator: Step 2

```

# Step3 [Prompt Generation]:
Create a clear and concise prompt that is optimized for {model_name} for {image editing/video
generation} while following the instructions below.

{Meta-Strategy}

## Strict Constraints:
- NO mention of texts, arrows, marks, or symbols overwritten on the image or point cloud (e.g.,
"moving toward the red marker").
- NO adding texts, arrows, marks, or symbols that were not in the original image (e.g., "add a
blue marker to the image").
- Avoid creating too long or too complex a prompt.

## Task:
Summarize the {category}(s)'s 3D movement and camera motion as a prompt while focusing on the
depth-wise final position/orientation relative to the initial state.
If there was no mention of camera motion, assume the camera is stationary. Use the surrounding
objects as reference points for your description.

```

Figure 13: User prompt template for the Translator: Step 3

```

# Step4 [Feedback-based Revision] (Loop):
{An image/A video} was generated using the prompt you just created, and a Verifier assessed
whether the generated {image/video} met the expected outcome. Revise the prompt based on the
feedback while following the instructions below.

## Verifier Feedback on the Generated {image/video}
"{feed_back}"

## Strict Constraints:
- NO mention of texts, arrows, marks, or symbols overwritten on the image or point cloud (e.g.,
"moving toward the red marker").
- NO adding texts, arrows, marks, or symbols that were not in the original image (e.g., "add a
blue marker to the image").
- Avoid creating too long or too complex a prompt.

## Task:
Fix only the problematic parts of the previous prompt based on the feedback, and create a prompt
that is optimized for {model_name} for {image editing/video generation}.
If there are no significant issues, please paraphrase and recreate the whole prompt while
preserving its intent.

```

Figure 14: User prompt template for the Translator: Step 4

```

# Role:
You are a strict verifier responsible for the verification of 3D-aware {image editing/video
generation} models.

```

Figure 15: System prompt template for the Verifier

```

# Step1 [Checklist Generation] Task:
Generate a comprehensive verification checklist to evaluate if a "Generated Image" (to be
provided in the next step) has successfully transformed the "Source Image" to match the "Target
Image".
The checklist must facilitate a three-way comparison between the Source Image, the Target Image,
and the Generated Image.

## Visual Contexts:
A concatenated image is provided for comparison:
- Left: The Source Image (Before editing)
- Right: The Target Image (Desired edited result)
The {category} undergoes a specific transformation (e.g., translation, rotation) in the Target
Image relative to the Source Image. Carefully identify the transformation and generate a
checklist accordingly.

## Expected Motion (from prompt or prior analysis, if any):
- {category}: {expected_subject_motion}

## Requirements
- The checklist must consist of at least 60 items.
- Include at least 5 negative constraints.

## Output checklist format:
*checklist*
1. <question 1>
2. <question 2>
3. ...

```

Figure 16: User prompt template for the Verifier: Step 1 (Image Editing)

```

# Step1 [Checklist Generation] Task:
Generate a comprehensive verification checklist to evaluate how well a "Generated Video" (to be
provided in the next step) matches the "Target Video", using the first frame as the reference.
The checklist must facilitate a three-way comparison between the first frame, the Target Video,
and the Generated Video, frame by frame, sequentially.

## Visual Contexts:
{num_frames} sequential video frames are provided:
- Key Frames from Target Video (Desired generated video)
The {category} undergoes a specific transformation (e.g., translation, rotation) in the Target
Video. Carefully identify the transformation and generate a checklist accordingly.

## Expected Motion (from prompt or prior analysis, if any):
- {category}: {expected_subject_motion}
- Camera: {expected_camera_motion}

## Requirements
- The checklist must consist of at least 60 items.
- Include at least 5 negative constraints.
- Include verification of {category}'s motion and trajectory, according to the expected motion
and the Target Video.
- Include verification of the camera motion and perspective trajectory, according to the
expected motion and the Target Video.

## Output checklist format:
1. <question 1>
2. <question 2>
3. ...

```

Figure 17: User prompt template for the Verifier: Step 1 (Video Generation)

```

# Step2 [Verification] Task:
Verify the generated image against the checklist by answering each checklist item.

## Visual Contexts:
A concatenated comparison is provided:
- Left: The Source Image (Before editing)
- Middle: The Target Image (Desired edited result)
- Right: The Generated Image (After editing)

## Output format:
*verification results*
1. <answer to the question 1> (If the answer is negative, provide a concise description (<= 20 words))
2. <answer to the question 2> (If the answer is negative, provide a concise description (<= 20 words))
3. ...

```

Figure 18: User prompt template for the Verifier: Step2 (Image Editing)

```

# Step2 [Verification] Task:
Verify the generated video against the checklist by answering each checklist item.

## Visual Contexts:
{num_frames} sequential video frames are provided.
- Key Frame from Generated Video (Actually generated video)

## Output format:
*verification results*
1. <answer to the question 1> (If the answer is negative, provide a concise description (<= 20 words))
2. <answer to the question 2> (If the answer is negative, provide a concise description (<= 20 words))
3. ...

```

Figure 19: User prompt template for the Verifier: Step2 (Video Generation)

```

# Step3 [Improvement Feedback] Task:
Provide improvement feedback on the prompt used for {image editing/video generation} based on the verification results.
Do not enumerate or quote checklist questions in the feedback.

## Prompt used for {image editing/video generation}:
{prompt}

```

Figure 20: User prompt template for the Verifier: Step3

```

# Task:
Analyze the provided JSON data, focusing on the motion description. Each sample has a "best"
pair (prompt that worked well and its feedback) and a "worst" pair (prompt that performed poorly
and its feedback). Based on these best vs worst examples, extract universal and generalized
principles for crafting effective {image editing/video generation} prompts.
**Set aside your usual assumptions, and focus your analysis on what can be directly inferred
from JSON data.**

# Output Format:
## Prompt Generation Guidelines

### 1. Good Motion Description Characteristics
* Identify key elements that lead to successful execution
* ...
### 2. Poor Motion Description Characteristics
* Identify common pitfalls that cause failure or AI confusion
* ...
### 3. Overall Characteristics of Best Prompts
* Optimal sentence length
* Writing style for maximum impact
* ...

# Data (best/worst pairs per record):
{
  "samples": ...
}

```

Figure 21: User prompt template for the History Summarization with an MLLM

```

# Prompt Examples (All Attempts) The following JSON contains, per record, all attempts with
prompt and feedback.
Each sample has "record_name" and "attempts": a list of { "attempt": N, "prompt": "...",
"feedback": "..." }.
Use these examples when generating or analyzing prompts.

```

Figure 22: Prompt header for history summarization without an MLLM using full histories

```

# Prompt Examples
The following is the best/worst prompt and feedback pairs per record (JSON).
Each sample has "best" (prompt that worked well and its feedback) and "worst" (prompt that
performed poorly and its feedback).
Please refer to these examples when generating your prompts.

```

Figure 23: Prompt header for history summarization without an MLLM using best/worst histories