Vision-as-Inverse-Graphics Agent via Interleaved Multimodal Reasoning

Shaofeng Yin[1,4*], Jiaxin Ge[1], Zora Zhiruo Wang[2], Chenyang Wang[1,4], Xiuyu Li[1]
Michael J. Black[3], Trevor Darrell[1], Angjoo Kanazawa[1], and Haiwen Feng[1,3,4†]

[1]University of California, Berkeley, USA [2]Carnegie Mellon University, USA
[3]Max Planck Institute for Intelligent Systems, Germany [4]Impossible, Inc., USA

[*]shaofeng.yin@impossible.place [†]haiwen.feng@berkeley.edu

fugtemypt123.github.io/VIGA-website

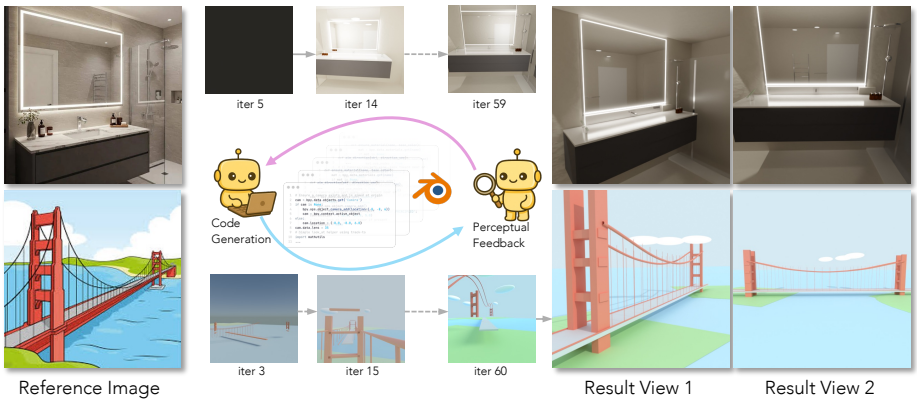

Fig. 1: VIGA constructs 3D scenes as executable programs from a reference image. Starting from an empty 3D environment, VIGA constructs the scene by leveraging abstract *symbolic generation* (writing and executing code to modify the scene) and active *visual verification* (rendering the state to inspect visual discrepancies and provide grounded feedback). By seamlessly interleaving code execution with fine-grained visual perception, the agent progressively refines the layout, geometry, and lighting to match the reference image.

Abstract. Vision-as-inverse-graphics, the concept of reconstructing images into editable programs, remains challenging for Vision-Language Models (VLMs), which inherently lack fine-grained spatial grounding in one-shot settings. To address this, we introduce VIGA (Vision-as-Inverse-Graphics Agent), an interleaved multimodal reasoning framework where symbolic logic and visual perception actively cross-verify each other. VIGA operates through a tightly coupled *code-render-inspect* loop: synthesizing symbolic programs, projecting them into visual states, and inspecting discrepancies to guide iterative edits. Equipped with high-level semantic skills and an evolving multimodal memory, VIGA sustains evidence-based modifications over long horizons. This training-free, task-agnostic framework seamlessly supports 2D document generation, 3D reconstruction,

multi-step 3D editing, and 4D physical interaction. Finally, we introduce BlenderBench, a challenging visual-to-code benchmark. Empirically, VIGA substantially improves accuracy compared with one-shot baselines in BlenderGym (*35.32%*), SlideBench (*117.17%*) and our proposed BlenderBench (*124.70%*).

1 Introduction

Consider the images in Fig. 1. What would it take to recreate these scenes symbolically in a graphics engine like Blender? Starting from scratch, we would need to write code that crafts or imports appropriate assets, establishes scene layout, places objects, and configures materials and lighting. Such an ability would yield compositional scene representations, easily manipulated for applications like robot training or interpretable digital twins. This capability represents achieving "vision-as-inverse-graphics", a longstanding aspiration since Larry Roberts' seminal 1963 thesis on the blocks world [38].

However, solving inverse graphics for complex scenes [52, 62] remains extremely difficult. Accurate reconstruction is not a passive, one-shot generation problem; it inherently requires continuous observation, active information-seeking, and precise geometric refinement. Existing approaches largely fall into two paradigms. On the one hand, differentiable inverse graphics [5, 11, 29, 30] provides fine-grained visual grounding through pixel-level optimization. However, its strict reliance on local gradients makes it vulnerable to local minima, especially when the discrepancy is large between the initial state and the target reference. Fundamentally lacking the capacity for high-level semantic reasoning and active exploration, the optimization process is unable to leap out of these non-convex traps. On the other hand, Vision-Language Models (VLMs) offer a complementary paradigm. Leveraging broad semantic priors, VLMs can generate scene code in a single pass, bypassing the local minima that trap gradient-based methods. However, this one-shot paradigm is inherently open-loop. Without fine-grained visual grounding, they struggle to precisely verify and refine complex scene details, such as camera extrinsics, object poses, and lighting. Consequently, even powerful proprietary VLMs struggle with one-shot code generation for accurate complex scene reconstruction [42].

In this work, we demonstrate that these limitations can be overcome by leveraging their complementary strengths. We present VIGA (Vision-as-Inverse-Graphics Agent), a multimodal coding agent that tightly couples discrete symbolic generation with continuous visual perception. By operating through a concise *code-render-inspect* loop, VIGA allows code logic and visual evidence to actively cross-verify each other. From a modern agentic perspective, this is a direct instantiation of interleaved multimodal reasoning. It aligns with the emerging notion of "thinking with images", but in a substantially richer setting: rather than relying on simple 2D image operations (*e.g.*, crop/zoom), VIGA reasons through a graphics engine whose action space is comprehensive, including camera poses, 3D geometry, materials, lighting, visibility, timestamps, etc. In a nutshell, this agent thinks with a graphics engine.

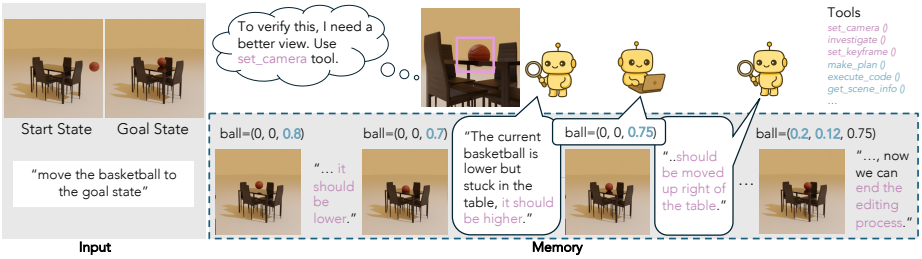

Fig. 2: The main pipeline of VIGA. VIGA operates through a continuous *code-render-inspect* loop. At each step, the agent synthesizes a program, which is executed to render a new scene. The agent then actively inspects this scene by invoking read-only perceptual interfaces to adjust camera viewpoints, identify the dominant discrepancy, and feed this visual feedback into the next step. We illustrate the scene editing task here. VIGA can also synthesize scenes from scratch given a goal specification (as in Fig. 1).

To robustly execute this multi-turn reasoning (Fig. 2), VIGA relies on two critical pillars. First, it equips the agent with a **high-level semantic skill library** that strictly decouples scene observation from modification. For instance, the agent can repeatedly invoke read-only perceptual interfaces (*e.g.*, `investigate`) to actively shift camera viewpoints and gather multi-angle visual evidence without altering the underlying scene. By explicitly comparing these rendered multi-angle views against the target image to localize discrepancies, the agent then triggers execution interfaces (*e.g.*, `execute_code`) to apply targeted programmatic edits. Second, it leverages an **evolving, sliding-window multimodal context memory** to execute evidence-based modifications. This structured memory retains recent plans, code diffs, and render histories. Crucially, its sliding-window mechanism actively bounds this retained history, inherently preventing context bloat and rot across long-horizon trajectories.

VIGA is a zero-shot, training-free framework. The verification process relies only on the agent's ability to interpret discrepancies between renders and reference images. Therefore it requires no auxiliary predictors or differentiable renderers. This design makes the approach exceptionally task and model-agnostic. It seamlessly generalizes across 2D document layout editing (Fig. 7), 3D scene reconstruction (Fig. 6), multi-step scene editing (Fig. 4), and 4D physical interaction (Fig. 8). Under a unified protocol, VIGA serves as a powerful capability testbed to evaluate heterogeneous foundation VLMs. To rigorously stress-test this unified paradigm, we introduce **BlenderBench**, a challenging suite of 27 tasks covering spatial adjustments, progressive editing, and compositional generation. VIGA delivers substantial gains across diverse benchmarks, yielding signficant improvements on BlenderGym (*+35.32%*), SlideBench (*+117.17%*), and BlenderBench (*+124.70%*).

In summary, we make the following contributions: (1) We introduce VIGA, a training-free, execution-grounded agent that reconstructs and edits scenes by closing the loop between code synthesis, rendering, and visual verification.

This reframes vision-as-inverse-graphics as a coding agent problem, where the rendered world becomes a medium for interleaved multimodal reasoning. (2) We propose a robust architectural paradigm combining a high-level semantic skill library with an evolving multimodal context memory, effectively sustaining long-horizon cross-modal reasoning without finetuning. (3) We demonstrate that VIGA delivers substantial empirical gains across a diverse spectrum of tasks, seamlessly supporting 2D document layout, 3D reconstruction, multi-step scene editing, and 4D physical interaction. (4) We release BlenderBench, a challenging, open-ended benchmark designed to rigorously stress-test agentic inverse graphics capabilities, where VIGA shows significant improvements against existing baselines.

2 Related Work

LLM-based Agent. LLM-based agents that observe and act in digital environments, such as browsers [60] and computers [54], have achieved impressive progress across various tasks, such as resolving issues in software engineering [17, 46, 58], or navigating through web pages to seek information or complete tasks on behalf of the users [3, 34, 67]. However, most agents behave effectively only when perceiving the environment as text and fall short when processing visual information, such as selecting products of certain styles on shopping websites [24]. These text-centric agents become even more limited when situated in visual environments such as 2D or 3D spaces [6, 7, 9, 16, 19, 23, 40, 43, 50, 57, 63], thus often fall short in producing high-quality visuals [13, 14]. In this work, we bridge this gap by presenting a unified agent that performs interleaved multimodal reasoning to iteratively synthesize and analyze visuals across 2D, 3D, and 4D spaces. We facilitate multi-turn verification and refinement through adaptive agent memory [44, 51, 56, 61, 64].

Visual Analysis or Synthesis with Code. Structured code has been used separately for image analysis and synthesis. On the analysis side, visual programming approaches [12, 16, 20, 39, 43] generate code to solve visual understanding tasks such as VQA. However, these approaches generate code in a single pass and do not edit the code based on any visual feedback. On the synthesis side, procedural content generation approaches [8, 35, 36] rely on predefined rules to generate specific 3D content. Recently, instruction-driven approaches generate visual content from text instructions leveraging large language models, spanning SVG generation [49, 53, 55, 59], slide generation [4, 13, 45, 65], and 3D scene generation [1, 10, 21, 27, 32, 33, 40–42, 66]. These approaches lack effective analysis that can ground the scene back to code, and thus fail to update the scene with code editing. We unify the two lines of work within a single framework of interleaved multimodal reasoning that executes programs, verifies generation, and then edits code in an iterative analysis-by-synthesis loop.

Vision-as-Inverse-Graphics through Programs. Since Larry Roberts's seminal Blocks World thesis [38], computer vision has often been framed as the inverse of computer graphics. A stricter interpretation of this perspective aims to recover a *graphics program* [15], *i.e.*, an interpretable, compositional abstraction

of a scene, from natural images using neural-symbolic methods [37]. For instance, PICTURE [26] introduced a probabilistic programming framework for representing arbitrary 2D and 3D scenes, demonstrating early successes on faces, bodies, and objects. Similarly, Wu *et al*. [52] proposed inferring structured markup code from images, which can be rendered directly. Although effective for multi-object abstract scenes like clip-art or Minecraft, their approach struggled to generalize to real images. More recent approaches leverage large language models (LLMs) to generate scene programs and refine them via supervision or verification feedback [14, 25, 27, 40]. While closely related to our work, these methods differ in two key respects. First, they rely on synthetic datasets or domain-specific verification pipelines, which restrict scalability and adaptability to diverse forms of visual feedback. Second, their workflows are typically predefined or tailored to particular downstream tasks, such as 3D scene reconstruction using pre-defined graphics engines and model weights.

In contrast, VIGA introduces a unified, training-free agent that seamlessly interleaves discrete symbolic code generation with active visual verification. While existing methods often rely on rigid, task-specific pipelines, our framework equips the agent with a versatile semantic skill library. The agent dynamically invokes these tools to probe the environment and execute changes, rather than being constrained by embedded, domain-specific architectures. By operating through an execution-grounded *code-render-inspect* loop sustained by an evolving multimodal context memory, VIGA inherently functions as a task-agnostic framework. This enables the agent to adapt zero-shot to a wide range of complex, long-horizon inverse graphics tasks—spanning from compositional 3D scene editing to dynamic 4D physical simulations—without requiring any model finetuning.

3 Method

Recognizing the complementary strengths of VLMs (for holistic semantic reasoning) and differentiable inverse graphics (for fine-grained visual refinement), we introduce VIGA, an interleaved multimodal reasoning framework designed to leverage both paradigms. Rather than treating reasoning steps in isolation, VIGA tightly couples these processes: the agent employs active visual navigation to systematically explore and overcome severe spatial misalignments (Fig. 3), while translating these fine-grained visual observations into targeted, programmatic code edits (Fig. 4). As illustrated in Fig. 2, VIGA operationalizes this continuous cross-modal loop via three core components. First, the agent drives an interactive reasoning paradigm across the visual and symbolic spaces (Sec. 3.1). To sustain this multi-turn trajectory, it leverages a dynamically evolving context memory (Sec. 3.2). Finally, it utilizes a high-level semantic skill library to reliably ground its perception and execution capabilities (Sec. 3.3).

3.1 Interleaved Multimodal Reasoning Framework

Formally, given an input target s_g^* (*e.g.*, a reference image) and an empty initial state s_0, the objective of inverse graphics is to synthesize a final executable

Algorithm 1 VIGA Interleaved Multimodal Reasoning Loop.

Require: Target specification s_g^*
Initialize contextual memory $M_0 \leftarrow \emptyset$
for $t = 0, 1, \ldots, T-1$ **do**
  // Phase 1: Symbolic Reasoning from Visual Input
  $\tau_t^{plan} \leftarrow \mathcal{A}(s_g^*, M_t)$ ▷ Formulates high-level plan
  if τ_t^{plan} includes `end_process` **then break**
  end if
  $p_t \leftarrow \mathcal{A}(s_g^*, M_t, \tau_t^{plan})$ ▷ Synthesizes executable program
  // Phase 2: Cross-Modal Execution
  $s_t \leftarrow \text{exec}(p_t)$ ▷ Renders concrete geometric states
  // Phase 3: Visual-to-Symbolic Feedback
  $\tau_t^{obs} \leftarrow \mathcal{A}(s_g^*, s_t, M_t)$ ▷ Explores the rendered scene
  $f_t \leftarrow \mathcal{A}(s_g^*, s_t, \tau_t^{obs}, M_t)$ ▷ Translates into actionable feedback
  // Phase 4: Context Memory Update (Sec. 3.2)
  $M_{t+1} \leftarrow \text{Tail}_L(M_t \cup \{p_t, f_t\})$
end for

Fig. 3: Agentic Visual Navigation. Demonstration of a visual inspection trajectory in a cluttered scene. The agent autonomously invokes spatial tools (focusing, zooming) to locate the inconspicuous target before evaluating attributes, showcasing an interleaved code-visual reasoning loop.

program p_T that renders a state s_T visually consistent with s_g^*. We implement this iterative analysis-by-synthesis process via interleaved multimodal reasoning. Specifically, we formulate this as a dynamic execution loop that fluidly transitions across different multimodal action spaces, utilizing each modality to cross-verify and inherently ground the others. Rather than naively chaining isolated single-modal steps, our approach deeply integrates these spaces: the agent leverages active visual navigation to verify spatial configurations and expose discrepancies (Fig. 3), which subsequently guide precise programmatic edits in the symbolic space (Fig. 4), forming a robust, self-correcting reasoning cycle.

Symbolic Reasoning from Visual Input. At each iteration t, the agent $\mathcal{A}$ drives the generation loop forward. Conditioned on its current context memory M_t, it evaluates the target s_g^* to derive a strategic planning trajectory $\tau_t^{plan} = \mathcal{A}(s_g^*, M_t)$. To instantiate this high-level plan, the agent must operate within the discrete symbolic space to synthesize exact programmatic actions. During this phase, $\mathcal{A}$ leverages forward anticipation, mentally mapping abstract symbolic parameters to their continuous visual consequences, to construct the valid code:

$$p_t = \mathcal{A}(s_g^*, M_t, \tau_t^{plan}). \tag{1}$$

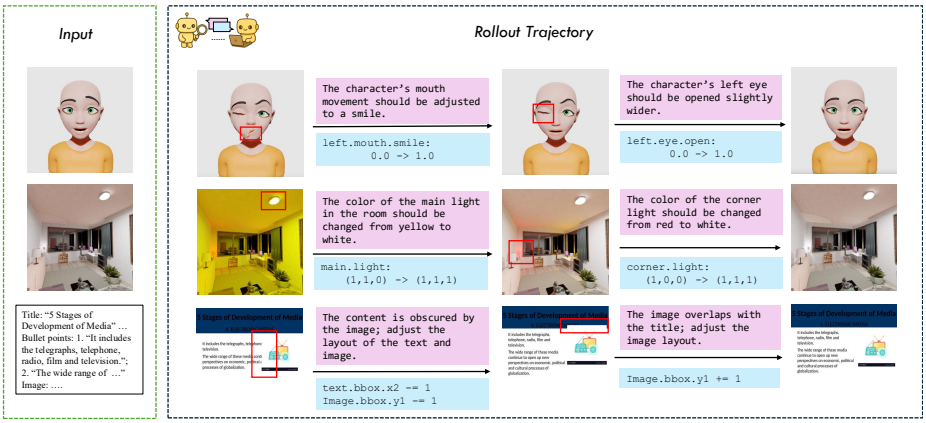

Fig. 4: Fine-Grained Visual Grounding. Agent trajectories across different tasks. The agent detects nuanced visual discrepancies (*e.g.*, mouth shape, lighting color) and dynamically maps these high-level observations directly to precise code parameters, avoiding rigid rule-based heuristics.

Cross-Modal Execution. Unlike standard coding tasks where the generated program p_t serves as a terminal textual artifact, our framework utilizes p_t to drive a deterministic modality shift. Instead of treating the code as an open-loop endpoint, we execute it within a 3D graphics engine to render a continuous visual state: $s_t = \text{exec}(p_t)$. This execution acts as the crucial cross-modal bridge. By projecting abstract symbolic structure into concrete visual observations, this mapping structurally defines the interaction loop: the symbolic syntax provides a precise action space for modifying the environment, while the newly rendered visual state immediately exposes spatial discrepancies, directly driving the next iteration of visual-to-code feedback.

Visual-to-Symbolic Feedback. To close the iterative loop, the agent seamlessly transitions into an active observation phase within the continuous visual space. By employing interactive tools (*e.g.*, adjusting camera viewpoints) to explore the newly rendered scene s_t, the agent $\mathcal{A}$ generates a comprehensive observation trajectory: $\tau_t^{obs} = \mathcal{A}(s_g^*, s_t, M_t)$. Finally, it rigorously compares this rendered outcome against the target specification s_g^*, translating any observed visual discrepancies into actionable natural language feedback. This feedback serves as the critical bridge back to the symbolic domain, guiding the agent's next iteration of programmatic edits:

$$f_t = \mathcal{A}(s_g^*, s_t, \tau_t^{obs}, M_t). \tag{2}$$

This grounded visual feedback f_t, alongside the synthesized program p_t, is subsequently assimilated to dynamically evolve the context memory into M_{t+1}, providing the critical foundation for the ensuing reasoning cycle detailed next.

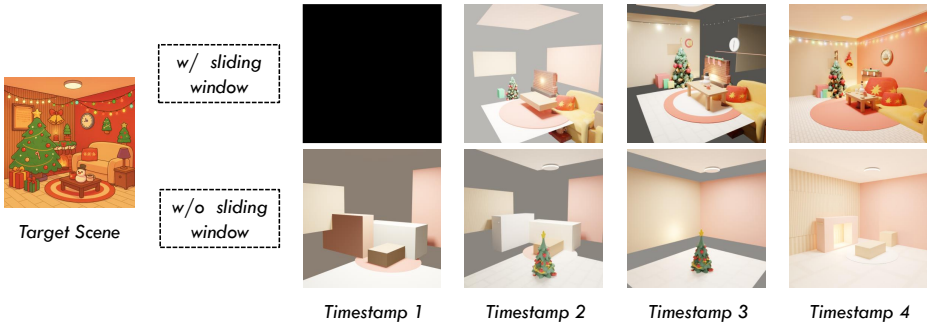

Fig. 5: Ablation on Evolving Multimodal Memory. We compare the sequential generation process with (w/) and without (w/o) our sliding window memory mechanism. With the memory window, the agent successfully maintains long-horizon context, progressively building a coherent scene (tree, sofa, fireplace). Without it, the agent suffers from severe context loss, losing previously generated objects (*e.g.*, tree) and spatial relationships in later iterations.

3.2 Evolving Multimodal Memory for Long-Horizon Reasoning

Unlike standard text-based chat logs, the memory M_t is constructed as a structured multimodal trajectory. At each iteration, it systematically archives the interleaved sequence of reasoning traces (plans and feedback f_t), symbolic actions (executable programs p_t), and grounded observations (rendered states s_t). This structured formulation is indispensable for fine-grained visual grounding. As illustrated in Fig. 2, maintaining this multimodal historical context empowers the agent to perform complex, progressive spatial refinements. By explicitly referencing past programmatic attempts alongside their resulting visual errors, the agent can systematically converge toward the exact target state. For instance, to rectify a spatial collision (*e.g.*, a basketball intersecting a table), the agent actively contrasts prior height parameters (0.8 vs. 0.7) to deduce the precise geometric configuration (0.75). Without this evolving memory, the agent is restricted to isolated, single-step edits, inevitably oscillating between incorrect states rather than converging on exact parameters.

Mitigating Context Bloat. While memory ensures continuous refinement, naively accumulating the complete multimodal trajectory introduces a critical bottleneck. Such unbounded expansion causes context bloat, scaling the computational overhead and introducing distracting noise—a phenomenon known as *context rot* [18, 28]. As the sequence lengthens, the foundation model's reasoning precision drastically degrades. To combat this, we constrain the memory using a fixed-size sliding window that retains only the most recent L iterations. Fig. 5 demonstrates this necessity: while our window-controlled memory maintains stable generation quality over long horizons, an unconstrained memory baseline rapidly collapses. Crucially, this aggressive truncation is inherently lossless. Because the latest synthesized program p_t functions as a self-contained symbolic

Semantic Skill	Encapsulated APIs	Capability
State Observation		
Spatial Navigation	`set_camera`, `investigate`	Actively adjust viewpoints to inspect regions and bypass occlusions.
Temporal Inspection	`set_keyframe`	Navigate the 4D timeline to evaluate dynamic physical motions.
Structural Query	`get_scene_info`, `set_visibility`	Query geometric attributes and toggle object visibility.
Diagnostic Rendering	`initialize_view`	Render canonical views and compute target bounding boxes.
State Modification		
Asset Instantiation	`init_scene`, `get_better_assets`	Retrieve or generate high-fidelity 3D assets via external modules to populate scenes.
State Execution	`execute_code`	Apply synthesized programs to persistently update scene states.
Trajectory Control	`undo_action`, `end_process`	Revert erroneous edits or cleanly terminate the reasoning loop.

Table 1: High-Level Skill Library as Semantic Interfaces. Rather than exposing raw graphics APIs, we equip the agent with encapsulated capabilities. The library is bifurcated into State Observation (gathering visual/structural context without altering the scene) and State Modification (executing persistent changes), mirroring our dual-space reasoning framework.

state that strictly encapsulates all prior successful geometric edits, it inherently renders older, intermediate reasoning traces redundant.

3.3 Skill Library as a Semantic Interface for Graphics

Action Space Compression. A naive approach to scene generation is to prompt the VLM to output raw, low-level graphics API calls (*e.g.*, raw Blender Python scripts). However, standard graphics engines feature a high-dimensional and brittle action space. Exposing the agent directly to this low-level syntax severely exacerbates context bloat and frequently leads to syntactic hallucinations or non-executable loops. To address this, we conceptualize our skill library not as rigid low-level APIs, but as a set of high-level semantic interfaces (Tab. 1). By encapsulating fragile, atomic operations into reliable, macro-level skills, we significantly compress the action space. This structural abstraction essentially insulates the agent from low-level programming trivialities, freeing its entire context window and cognitive capacity for what it excels at: high-level multimodal reasoning and abstract planning [2].

Perception, Reasoning, and Action. To seamlessly bridge the symbolic and visual modalities, the agent operates through a continuous triad of perception, reasoning, and action. At the core of this loop is the agent's internal reasoning capacity, which cautiously orchestrates the external skill library (Tab. 1). Rather than blindly manipulating the environment, the agent first relies on *State Observation* skills (*e.g.*, `investigate` and `get_scene_info`). Because these interfaces are inherently read-only and non-destructive, the agent is free to conduct exhaustive, multi-turn spatial queries to achieve fine-grained visual grounding. Crucially, this unconstrained exploration facilitates a cognitive leap from visual appearance to structural abstraction. It allows the VLM to distill raw visual

phenomena into a high-level semantic understanding of the scene's underlying physical, geometric, and spatial properties. Only after this rigorous transition from surface-level grounding to abstract reasoning does the agent trigger a persistent change via *State Modification* skills. Central to this is `execute_code`, which translates this semantic understanding into exact parametric adjustments by injecting the synthesized program (p_t) into the engine. By strictly decoupling fine-grained visual perception from abstract code execution, the framework ensures that scene mutations are conditioned on a comprehensive series of prior observations and are highly deliberate.

4 Experiments

We evaluate VIGA across a diverse set of visual generation and editing tasks to demonstrate the effectiveness of its interleaved multimodal reasoning paradigm. We start by presenting qualitative results that highlight the visual fidelity and broad adaptability of VIGA across 2D, 3D, and 4D domains (Sec. 4.1). Recognizing that traditional benchmarks primarily assess static, single-turn outputs, we introduce BlenderBench, a challenging new benchmark designed to stress-test multi-step, dynamic agentic behaviors (Sec. 4.2). Furthermore, we demonstrate the generalization capability of VIGA, achieving substantial performance gains on established quantitative benchmarks (Sec. 4.3).

4.1 Qualitative Results across Diverse Tasks

We present qualitative results across 3D scene reconstruction, 4D physics simulation, and 2D document design to demonstrate the generation capabilities and broad adaptability of VIGA. These examples illustrate that a unified interleaved reasoning loop naturally generalizes to diverse visual tasks without relying on domain-specific architectures or heuristics.

3D Generation and Editing. As illustrated in Fig. 6, VIGA reconstructs diverse 3D scenes from sparse single-view inputs, ranging from photorealistic images to hand-drawn sketches. When populating scenes spanning from object-centric arrangements to complex indoor environments (*e.g.*, sushi, bedroom), the agent dynamically invokes the `get_better_assets` tool to retrieve and position high-quality external meshes. It then iteratively optimizes the spatial layout and material properties of these assets to faithfully reproduce the target's lighting and textures. Beyond parsing natural images, VIGA exhibits strong sketch-to-3D generalization, converting sparse line drawings (*e.g.*, an architectural house sketch) into complete 3D geometries.

4D Dynamic Scene Simulation. Beyond static reconstruction, VIGA simulates 4D physical dynamics given a single image and a text prompt (Fig. 8). By inferring and configuring underlying physical parameters (*e.g.*, mass and collision shapes), the agent translates static scenes into complex interactive environments, such as a ball scattering fruit, a mirror shattering into reflective shards, or an earthquake disrupting tabletop objects. While maintaining consistent reflections on moving

Fig. 6: Qualitative Results on 3D Scene Generation. VIGA accurately generates high-fidelity 3D scenes from diverse visual inputs with precise semantic and visual alignment.

Fig. 7: Qualitative Results on 2D Document Design. VIGA generates high-quality presentation slides from text instructions.

glass or resolving multi-object collisions remains a fundamental bottleneck for pixel-based generative models, VIGA bypasses these limitations. By executing programmatic commands within a physics engine, the framework inherently enforces strict physical accuracy and temporal consistency.

2D Document Design. In addition to 3D and 4D spatial graphics, Fig. 7 demonstrates the framework's applicability to 2D document generation. By substituting the underlying execution engine, VIGA synthesizes professional PowerPoint slides from scratch, orchestrating layout planning, text formatting, and image placement. This flexibility underscores a core advantage of our approach: interleaved multimodal reasoning naturally generalizes to any visual domain governed by a programmatic interface, utilizing this structured action space to iteratively converge on the target state.

4.2 BlenderBench: Evaluating Agentic Behaviors

Inspired by prior works on single-step graphics editing (*e.g.*, BlenderGym [14]), we introduce BlenderBench, a new benchmark designed to stress-test interleaved multimodal reasoning. As illustrated in Fig. 9, while traditional benchmarks primarily evaluate static final outputs, BlenderBench explicitly quantifies multi-step, dynamic agentic behaviors, such as active spatial exploration, iterative scene reasoning for attribute modification, and complex exploratory editing.

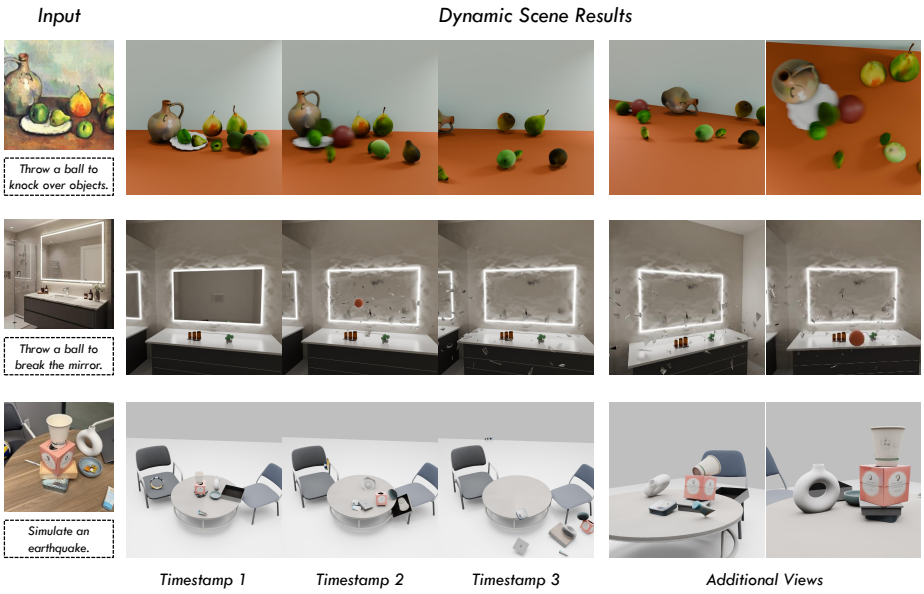

Fig. 8: Qualitative Results on 4D Dynamic Scene Simulation. Given a single reference image and a text instruction, VIGA accurately synthesizes complex physical dynamics (*e.g.*, multi-object collisions and glass shattering). It adaptively constructs scenes by fetching SAM-3D [47] assets via `init_scene` (*e.g.*, fruit, desk) or generating them programmatically from primitives (*e.g.*, mirror).

Setup. To stress-test interleaved multimodal reasoning, BlenderBench comprises three progressive tracks: Task 1 (Camera Adjustment), Task 2 (Fixed-View Editing), and Task 3 (Exploratory Editing). Because target references frequently incorporate domain gaps (*e.g.*, style transfers), we evaluate generation quality using Photometric Loss (PL) and Negative-CLIP Score (N-CLIP), and introduce a *VLM Score* to prioritize structural layout over task-irrelevant stylistic artifacts. Finally, we compare VIGA against a standard one-shot approach and the memory-less baseline BlenderAlchemy [22]. To ensure fair iterative comparisons, we adopt a *best-of-N* setting [14] to analyze the impact of search budgets. In this setting, the agent generates N candidate edits at each step and selects the optimal one for the next round.

Results. As shown in Tab. 2, VIGA consistently surpasses existing baselines, yielding an average relative improvement (Impr.) of +124.70% across all evaluation tracks and model settings. In Task 1 (Camera Adjustment), standard one-shot methods fail due to limited zero-shot spatial perception, whereas our agent dynamically updates its viewpoint to locate targets. Notably, scaling the *best-of-N* search budget from $N = 1$ to $N = 4$ under GPT-4o boosts the VLM Score from 1.44 to 3.25, effectively overcoming local minima during viewpoint alignment. Furthermore, for joint spatial-visual reasoning in Task 3 (Exploratory Editing),

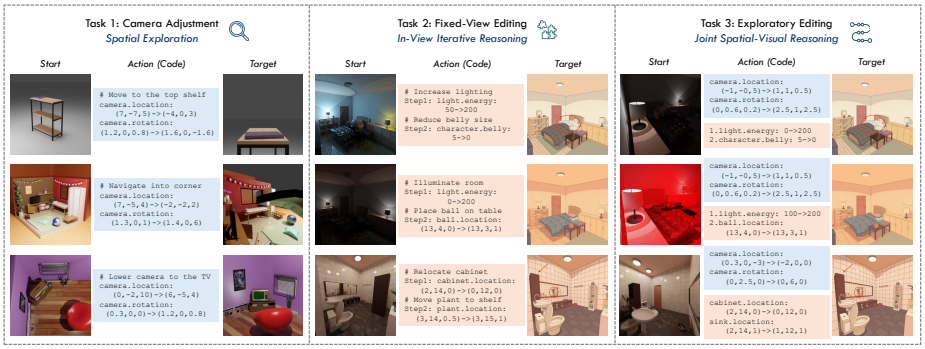

Fig. 9: Overview of the BlenderBench evaluation tracks. The benchmark comprises three progressive scenarios to stress-test interleaved multimodal reasoning. *Task 1 (Camera Adjustment)* isolates active spatial exploration for viewpoint alignment. *Task 2 (Fixed-View Editing)* evaluates iterative scene reasoning and attribute modification under a fixed camera pose. *Task 3 (Exploratory Editing)* unifies these challenges, requiring the agent to actively explore the spatial layout while simultaneously executing programmatic multi-step edits.

Model	Setting	Task 1			Task 2			Task 3			Impr. (%)
		PL ↓	N-CLIP ↓	VLM ↑	PL ↓	N-CLIP ↓	VLM ↑	PL ↓	N-CLIP ↓	VLM ↑	
GPT-4o	One-Shot	48.16	64.17	0.58	7.36	7.12	2.75	30.14	38.69	0.25	-
	BlenderAlchemy (best-of-1)	10.62	**17.68**	**1.75**	6.13	7.71	3.08	19.60	26.10	0.67	67.56
	VIGA (best-of-1)	**8.56**	18.19	1.44	**5.11**	**4.33**	**3.58**	**14.51**	**17.91**	**1.53**	**113.96**
		(+19.40%)	(-2.89%)	(-17.71%)	(+16.64%)	(+43.84%)	(+16.23%)	(+25.97%)	(+31.40%)	(+128.36%)	(+68.69%)
	BlenderAlchemy (best-of-4)	14.50	19.57	1.75	**1.95**	**2.47**	3.53	20.62	25.79	0.56	77.48
	VIGA (best-of-4)	**5.47**	**6.10**	**3.25**	2.94	3.50	**3.83**	**12.62**	**22.84**	**1.61**	**159.19**
		(+62.28%)	(+68.84%)	(+85.71%)	(-50.77%)	(-41.70%)	(+8.50%)	(+38.79%)	(+11.43%)	(+187.50%)	(+105.45%)
Qwen3-VL-8B	One-Shot	60.82	78.16	0.28	33.14	37.51	1.61	22.81	22.98	1.25	-
	BlenderAlchemy (best-of-1)	25.61	37.93	0.36	10.69	13.92	2.64	30.45	23.91	0.64	27.36
	VIGA (best-of-1)	**10.54**	**17.27**	**1.31**	**3.85**	**3.34**	**3.33**	**11.35**	**6.86**	**2.25**	**112.79**
		(+58.86%)	(+54.45%)	(+263.89%)	(+63.98%)	(+76.06%)	(+26.14%)	(+62.75%)	(+71.33%)	(+251.56%)	(+312.20%)
	BlenderAlchemy (best-of-4)	11.57	17.58	1.19	5.38	4.64	2.94	23.62	15.67	0.93	82.24
	VIGA (best-of-4)	**8.80**	**15.67**	**1.38**	**5.02**	**3.86**	**3.08**	**9.08**	**6.94**	**2.02**	**112.87**
		(+23.92%)	(+10.88%)	(+15.97%)	(+6.69%)	(+16.81%)	(+4.76%)	(+61.54%)	(+55.73%)	(+117.20%)	(+37.20%)

Table 2: Evaluation on BlenderBench. We report Photometric Loss (PL), Negative-CLIP Score (N-CLIP), and VLM Score (VLM).

BlenderAlchemy suffers from cross-step inconsistency, frequently overwriting earlier edits. By explicitly tracking states, VIGA executes complex modifications while strictly maintaining sequential consistency, achieving a +187.50% VLM Score improvement (GPT-4o, best-of-4). Beyond absolute metrics, the iterative reasoning loop acts as a critical scaffold across varying model capacities. When instantiated with Qwen, VIGA delivers a +312.20% relative gain over BlenderAlchemy (best-of-1), effectively bridging the performance gap between lightweight open-source models and larger closed-source models.

4.3 Evaluation on Existing Benchmarks

While primarily developed for interleaved multimodal reasoning, VIGA naturally generalizes to standard single-step programmatic generation and editing tasks.

Model	Setting	Shape		Place		Geom		Light		Mat		Impr. (%)
		PL ↓	NC ↓	PL ↓	NC ↓	PL ↓	NC ↓	PL ↓	NC ↓	PL ↓	NC ↓	
GPT-4o	One-Shot	7.94	19.96	11.86	29.31	18.12	24.91	2.06	2.17	8.78	15.25	-
	B-Alchemy (bo1)	7.82 (+14.3%)	20.19 (+17.3%)	11.57 (+14.5%)	28.07 (+4.3%)	15.58 (+24.1%)	20.61 (+23.5%)	1.44 (+0.7%)	1.82 (+22.5%)	7.22 (+15.4%)	12.05 (+22.1%)	12.33 (+118.1%)
	VIGA (bo1)	**6.70** (+14.3%)	**16.70** (+17.3%)	**9.89** (+14.5%)	**26.87** (+4.3%)	**11.82** (+24.1%)	**15.76** (+23.5%)	**1.43** (+0.7%)	**1.41** (+22.5%)	**6.11** (+15.4%)	**9.39** (+22.1%)	**26.88** (+118.1%)
	B-Alchemy (bo4)	5.78 (+5.7%)	17.19 (+23.9%)	10.29 (+5.1%)	25.58 (+31.7%)	9.47 (+22.1%)	11.68 (+37.0%)	0.97 (+27.8%)	1.23 (+17.9%)	9.47 (+67.9%)	12.20 (+37.5%)	27.63 (+76.8%)
	VIGA (bo4)	**5.45** (+5.7%)	**13.08** (+23.9%)	**9.76** (+5.1%)	**17.48** (+31.7%)	**7.38** (+22.1%)	**7.36** (+37.0%)	**0.70** (+27.8%)	**1.01** (+17.9%)	**3.04** (+67.9%)	**7.63** (+37.5%)	**48.86** (+76.8%)
Qwen3-8B	One-Shot	7.76	21.78	14.10	41.44	11.65	14.02	12.12	12.30	9.83	17.16	-
	B-Alchemy (bo1)	**12.11** (-11.6%)	31.50 (+14.1%)	13.86 (+27.3%)	39.47 (+28.8%)	9.62 (+4.9%)	14.47 (+15.4%)	5.22 (+55.4%)	6.44 (+59.9%)	6.62 (+2.9%)	16.98 (+32.1%)	5.81 (+289.8%)
	VIGA (bo1)	13.51 (-11.6%)	**27.07** (+14.1%)	**10.07** (+27.3%)	**28.12** (+28.8%)	**9.15** (+4.9%)	**12.24** (+15.4%)	**2.33** (+55.4%)	**2.58** (+59.9%)	**6.43** (+2.9%)	**11.53** (+32.1%)	**22.64** (+289.8%)
	B-Alchemy (bo4)	9.20 (+21.7%)	19.60 (+8.3%)	13.07 (+24.1%)	36.34 (+33.2%)	9.24 (+6.0%)	12.62 (+20.1%)	3.68 (+48.6%)	4.23 (+40.2%)	4.42 (+30.8%)	17.12 (+45.7%)	23.23 (+84.6%)
	VIGA (bo4)	**7.20** (+21.7%)	**17.97** (+8.3%)	**9.92** (+24.1%)	**24.28** (+33.2%)	**8.69** (+6.0%)	**10.08** (+20.1%)	**1.89** (+48.6%)	**2.53** (+40.2%)	**3.06** (+30.8%)	**9.30** (+45.7%)	**42.88** (+84.6%)

Table 3: Evaluation on BlenderGym. We report Photometric Loss (PL) and Negative-CLIP Score (NC). Here bo$N$ denotes the best-of-N setting.

Model	Setting	Exec. ↑	Qual. ↑	Over. ↑
GPT-4o	One-Shot	0.92	60.7	55.8
	VIGA	**0.95** (+3.3%)	**61.4** (+1.2%)	**58.3** (+4.5%)
Claude-3.5	One-Shot	**0.93**	65.3	60.7
	VIGA	0.90 (-3.2%)	**69.2** (+6.0%)	**62.2** (+2.5%)
Gemini-2.5	One-Shot	0.61	67.9	41.4
	VIGA	**0.69** (+13.1%)	**70.9** (+4.4%)	**48.9** (+18.1%)
Qwen3-8B	One-Shot	0.10	60.5	6.0
	VIGA	**0.54** (+440%)	**60.8** (+0.5%)	**32.8** (+446.7%)

Table 4: Evaluation on SlideBench. We report Execution, Quality, and Overall scores.

To demonstrate this strong task-agnostic capability, we evaluate the framework across two established benchmarks from distinct visual domains.

3D Editing (BlenderGym). We first evaluate VIGA on BlenderGym [14] for single-step 3D scene editing. As reported in Tab. 3, VIGA consistently surpasses both the one-shot baseline and the memory-less baseline BlenderAlchemy [22]. VIGA yields an average improvement (Impr.) of +35.32% across all the settings. These results indicate that VIGA precisely localizes and modifies fine-grained object attributes (*e.g.*, blend shapes, placement, and lighting) while strictly preserving the unedited elements of the scene.

2D Design (SlideBench). Extending to 2D environments, we evaluate VIGA on SlideBench [13] for programmatic PowerPoint generation. As reported in Tab. 4, VIGA delivers an average Overall score improvement of +117.17% across a diverse spectrum of foundation models, spanning open-source (*e.g.*, Qwen3-VL-8B) and closed-source (*e.g.*, GPT-4o, Claude-Sonnet-4) models. While standard one-shot methods frequently halt upon encountering a single syntax or rendering error, our interleaved reasoning loop acts as a robust error-recovery mechanism. By iteratively parsing execution logs to rectify faulty code, VIGA drastically improves the execution success rate (Exec.), ensuring the reliable generation of complex document layouts.

5 Conclusion

We present VIGA, a multimodal agent that tackles inverse graphics through interleaved multimodal reasoning. By deeply coupling discrete program synthesis with active visual verification, VIGA maps abstract spatial concepts to precise, executable code across 2D, 3D, and 4D domains. To systematically assess these dynamic agentic behaviors, we introduce BlenderBench, a comprehensive benchmark designed to quantify multi-step spatial reasoning and progressive scene editing. While current performance is bounded by the spatial perception capabilities of underlying VLMs and context window constraints in extremely long sequences, our framework is inherently extensible. As foundation models and external generative tools (*e.g.*, Meshy [31], Tripo [48], SAM-3D [47]) advance,

their capabilities will seamlessly plug into our closed-loop paradigm. Ultimately, by orchestrating an evolving memory and a semantic skill library into a unified, training-free agent, VIGA tackles complex, long-horizon inverse graphics, serving as a foundational step toward fully automated scene generation.

References

1. Aguina-Kang, R., Gumin, M., Han, D.H., Morris, S., Yoo, S.J., Ganeshan, A., Jones, R.K., Wei, Q.A., Fu, K., Ritchie, D.: Open-universe indoor scene generation using llm program synthesis and uncurated object databases (2024), https://arxiv.org/abs/2403.09675
2. Anthropic: The complete guide to building skills for claude. Tech. rep., Anthropic (jan 2026), https://resources.anthropic.com/hubfs/The-Complete-Guide-to-Building-Skill-for-Claude.pdf
3. Azam, R., Vempaty, A., Jagmohan, A.: Reflection-based memory for web navigation agents (2025), https://arxiv.org/abs/2506.02158
4. Bandyopadhyay, S., Maheshwari, H., Natarajan, A., Saxena, A.: Enhancing presentation slide generation by LLMs with a multi-staged end-to-end approach. In: Mahamood, S., Minh, N.L., Ippolito, D. (eds.) Proceedings of the 17th International Natural Language Generation Conference. pp. 222–229. Association for Computational Linguistics, Tokyo, Japan (Sep 2024). https://doi.org/10.18653/v1/2024.inlg-main.18
5. Blanz, V., Vetter, T.: A morphable model for the synthesis of 3d faces. Seminal Graphics Papers: Pushing the Boundaries, Volume 2 (1999), https://api.semanticscholar.org/CorpusID:203705211
6. Chen, B., Xu, Z., Kirmani, S., Ichter, B., Sadigh, D., Guibas, L., Xia, F.: Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14455–14465 (June 2024)
7. Cheng, A.C., Yin, H., Fu, Y., Guo, Q., Yang, R., Kautz, J., Wang, X., Liu, S.: Spatialrgpt: Grounded spatial reasoning in vision-language models. In: NeurIPS (2024)
8. Deitke, M., VanderBilt, E., Herrasti, A., Weihs, L., Salvador, J., Ehsani, K., Han, W., Kolve, E., Farhadi, A., Kembhavi, A., Mottaghi, R.: ProcTHOR: Large-Scale Embodied AI Using Procedural Generation. In: NeurIPS (2022), outstanding Paper Award
9. Dwedari, M.M., Niessner, M., Chen, Z.: Generating context-aware natural answers for questions in 3d scenes. In: Proceedings of the British Machine Vision Conference (BMVC). BMVA Press (2023)
10. Feng, W., Zhu, W., Fu, T.j., Jampani, V., Akula, A., He, X., Basu, S., Wang, X.E., Wang, W.Y.: Layoutgpt: compositional visual planning and generation with large language models. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. NIPS '23 (2023)
11. Feng, Y., Feng, H., Black, M.J., Bolkart, T.: Learning an animatable detailed 3D face model from in-the-wild images. ACM Transactions on Graphics (ToG), Proc. SIGGRAPH **40**(4), 88:1–88:13 (Aug 2021)

12. Ge, J., Subramanian, S., Shi, B., Herzig, R., Darrell, T.: Recursive visual programming. In: European Conference on Computer Vision. pp. 1–18. Springer (2024)
13. Ge, J., Wang, Z.Z., Zhou, X., Peng, Y.H., Subramanian, S., Tan, Q., Sap, M., Suhr, A., Fried, D., Neubig, G., et al.: Autopresent: Designing structured visuals from scratch. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 2902–2911 (2025)
14. Gu, Y., Huang, I., Je, J., Yang, G., Guibas, L.: Blendergym: Benchmarking foundational model systems for graphics editing. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 18574–18583 (2025)
15. Gulwani, S., Polozov, O., Singh, R.: Program synthesis. Found. Trends Program. Lang. **4**(1–2), 1–119 (2017). https://doi.org/10.1561/2500000010
16. Gupta, T., Kembhavi, A.: Visual programming: Compositional visual reasoning without training. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14953–14962 (2023)
17. He, J., Treude, C., Lo, D.: Llm-based multi-agent systems for software engineering: Literature review, vision, and the road ahead. ACM Trans. Softw. Eng. Methodol. **34**(5) (May 2025). https://doi.org/10.1145/3712003
18. Hong, K., Troynikov, A., Huber, J.: Context rot: How increasing input tokens impacts llm performance. Tech. rep., Chroma (July 2025), https://research.trychroma.com/context-rot, published July 2025.
19. Hong, Y., Zhen, H., Chen, P., Zheng, S., Du, Y., Chen, Z., Gan, C.: 3dllm: Injecting the 3d world into large language models. In: Advances in Neural Information Processing Systems (NeurIPS). vol. 36, pp. 20482–20494. Curran Associates, Inc. (2023)
20. Hu, Y., Stretcu, O., Lu, C.T., Viswanathan, K., Hata, K., Luo, E., Krishna, R., Fuxman, A.: Visual program distillation: Distilling tools and programmatic reasoning into vision-language models (2023)
21. Hu, Z., Iscen, A., Jain, A., Kipf, T., Yue, Y., Ross, D.A., Schmid, C., Fathi, A.: Scenecraft: An llm agent for synthesizing 3d scenes as blender code. In: Forty-first International Conference on Machine Learning (2024)
22. Huang, I., Yang, G., Guibas, L.: Blenderalchemy: Editing 3d graphics with vision-language models. In: European Conference on Computer Vision. pp. 297–314. Springer (2024)
23. Kodnongbua, M., Jones, B., Ahmad, M.B.S., Kim, V., Schulz, A.: Reparamcad: Zero-shot cad re-parameterization for interactive manipulation. In: ACM SIGGRAPH Asia 2023 Conference Papers. ACM, New York, NY, USA (2023). https://doi.org/10.1145/3610548.3618219
24. Koh, J.Y., Lo, R., Jang, L., Duvvur, V., Lim, M.C., Huang, P.Y., Neubig, G., Zhou, S., Salakhutdinov, R., Fried, D.: Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In: ICLR 2024 Workshop on Large Language Model (LLM) Agents (2024), https://openreview.net/forum?id=RPKxrKTJbj
25. Kulits, P., Feng, H., Liu, W., Abrevaya, V.F., Black, M.J.: Re-thinking inverse graphics with large language models. Transactions on Machine Learning Research (2024)

26. Kulkarni, T.D., Kohli, P., Tenenbaum, J.B., Mansinghka, V.: Picture: A probabilistic programming language for scene perception. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society (June 2015)
27. Ling, L., Lin, C.H., Lin, T.Y., Ding, Y., Zeng, Y., Sheng, Y., Ge, Y., Liu, M.Y., Bera, A., Li, Z.: Scenethesis: A language and vision agentic framework for 3d scene generation. arXiv preprint arXiv:2505.02836 (2025)
28. Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P.: Lost in the middle: How language models use long contexts. Transactions of the Association for Computational Linguistics **12**, 157–173 (2024). https://doi.org/10.1162/tacl_a_00638
29. Liu, S., Li, T., Chen, W., Li, H.: Soft rasterizer: A differentiable renderer for image-based 3d reasoning. 2019 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 7707–7716 (2019), https://api.semanticscholar.org/CorpusID:102484000
30. Loper, M., Black, M.J.: Opendr: An approximate differentiable renderer. In: European Conference on Computer Vision (2014), https://api.semanticscholar.org/CorpusID:17868098
31. Meshy: Meshy: Fast 3d generative ai. https://www.meshy.ai/ (2024)
32. Öcal, B.M., Tatarchenko, M., Karaoğlu, S., Gevers, T.: Sceneteller: Language-to-3d scene generation. In: Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXXV. pp. 362–378 (2024). https://doi.org/10.1007/978-3-031-73013-9_21
33. Qin, Y., Xu, Z., Liu, Y.: Apply hierarchical-chain-of-generation to complex attributes text-to-3d generation. 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 18521–18530 (2025), https://api.semanticscholar.org/CorpusID:278481349
34. Qin, Y., Ye, Y., Fang, J., Wang, H., Liang, S., Tian, S., Zhang, J., Li, J., Li, Y., Huang, S., et al.: Ui-tars: Pioneering automated gui interaction with native agents. arXiv preprint arXiv:2501.12326 (2025)
35. Raistrick, A., Lipson, L., Ma, Z., Mei, L., Wang, M., Zuo, Y., Kayan, K., Wen, H., Han, B., Wang, Y., Newell, A., Law, H., Goyal, A., Yang, K., Deng, J.: Infinite photorealistic worlds using procedural generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12630–12641 (2023)
36. Raistrick, A., Mei, L., Kayan, K., Yan, D., Zuo, Y., Han, B., Wen, H., Parakh, M., Alexandropoulos, S., Lipson, L., Ma, Z., Deng, J.: Infinigen indoors: Photorealistic indoor scenes using procedural generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21783–21794 (June 2024)
37. Ritchie, D., Guerrero, P., Jones, R.K., Mitra, N.J., Schulz, A., Willis, K.D.D., Wu, J.: Neurosymbolic models for computer graphics. Computer Graphics Forum **42**(2), 545–568 (2023), https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14775

38. Roberts, L.G.: Machine Perception of Three-Dimensional Solids. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA (1963)
39. Subramanian, S., Narasimhan, M., Khangaonkar, K., Yang, K., Nagrani, A., Schmid, C., Zeng, A., Darrell, T., Klein, D.: Modular visual question answering via code generation. arXiv preprint arXiv:2306.05392 (2023)
40. Sun, C., Han, J., Deng, W., Wang, X., Qin, Z., Gould, S.: 3d-gpt: Procedural 3d modeling with large language models. In: 2025 International Conference on 3D Vision (3DV). pp. 1253–1263. IEEE (2025)
41. Sun, F.Y., Liu, W., Gu, S., Lim, D., Bhat, G., Tombari, F., Li, M., Haber, N., Wu, J.: Layoutvlm: Differentiable optimization of 3d layout via vision-language models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 29469–29478 (2025)
42. Sun, F.Y., Wu, S., Jacobsen, C., Yim, T., Zou, H., Zook, A., Li, S., Chou, Y.H., Can, E., Wu, X., Eppner, C., Blukis, V., Tremblay, J., Wu, J., Birchfield, S., Haber, N.: 3d-generalist: Self-improving vision-language-action models for crafting 3d worlds (2025), https://arxiv.org/abs/2507.06484
43. Surís, D., Menon, S., Vondrick, C.: Vipergpt: Visual inference via python execution for reasoning. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 11888–11898 (2023)
44. Suzgun, M., Yuksekgonul, M., Bianchi, F., Jurafsky, D., Zou, J.: Dynamic cheatsheet: Test-time learning with adaptive memory. arXiv preprint arXiv:2504.07952 (2025)
45. Tang, W., Xiao, J., Jiang, W., Xiao, X., Wang, Y., Tang, X., Li, Q., Ma, Y., Liu, J., Tang, S., Lyu, M.R.: Slidecoder: Layout-aware rag-enhanced hierarchical slide generation from design (2025), https://arxiv.org/abs/2506.07964
46. Tawosi, V., Ramani, K., Alamir, S., Liu, X.: Almas: an autonomous llm-based multi-agent software engineering framework (2025), https://arxiv.org/abs/2510.03463
47. Team, S.D., Chen, X., Chu, F.J., Gleize, P., Liang, K.J., Sax, A., Tang, H., Wang, W., Guo, M., Hardin, T., Li, X., Lin, A., Liu, J., Ma, Z., Sagar, A., Song, B., Wang, X., Yang, J., Zhang, B., Dollár, P., Gkioxari, G., Feiszli, M., Malik, J.: Sam 3d: 3dfy anything in images. arXiv preprint arXiv:2511.16624v1 (2025), https://arxiv.org/abs/2511.16624v1, version 1, submitted November 20, 2025.
48. Tripo AI: Tripo: Ai 3d model generator. https://www.tripo3d.ai/ (2024)
49. Wang, F., Zhao, Z., Liu, Y., Zhang, D., Gao, J., Sun, H., Li, X.: Svgen: Interpretable vector graphics generation with large language models. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 9608–9617. MM '25, Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3746027.3755011
50. Wang, J., Ming, Y., Shi, Z., Vineet, V., Wang, X., Li, Y., Joshi, N.: Is a picture worth a thousand words? delving into spatial reasoning for vision language models. In: Proceedings of the 38th International Conference on Neural Information Processing Systems. NIPS '24 (2024)

51. Wang, Z.Z., Mao, J., Fried, D., Neubig, G.: Agent workflow memory. In: Forty-second International Conference on Machine Learning (2025), https://openreview.net/forum?id=NTAhi2JEEE
52. Wu, J., Tenenbaum, J.B., Kohli, P.: Neural scene de-rendering. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
53. Wu, R., Su, W., Liao, J.: Chat2svg: Vector graphics generation with large language models and image diffusion models. 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 23690–23700 (2024), https://api.semanticscholar.org/CorpusID:274280554
54. Xie, T., Zhang, D., Chen, J., Li, X., Zhao, S., Cao, R., Hua, T.J., Cheng, Z., Shin, D., Lei, F., Liu, Y., Xu, Y., Zhou, S., Savarese, S., Xiong, C., Zhong, V., Yu, T.: OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In: The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track (2024), https://openreview.net/forum?id=tN61DTr4Ed
55. Xing, X., Hu, J., Liang, G., Zhang, J., Xu, D., Yu, Q.: Empowering llms to understand and generate complex vector graphics. arXiv preprint arXiv:2412.11102 (2024)
56. Xu, W., Liang, Z., Mei, K., Gao, H., Tan, J., Zhang, Y.: A-mem: Agentic memory for llm agents. In: Advances in Neural Information Processing Systems (2025)
57. Xue, L., Gao, M., Xing, C., Martín-Martín, R., Wu, J., Xiong, C., Xu, R., Niebles, J.C., Savarese, S.: Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1179–1189. IEEE Computer Society (2023)
58. Yang, J., Jimenez, C.E., Wettig, A., Lieret, K., Yao, S., Narasimhan, K., Press, O.: Swe-agent: Agent-computer interfaces enable automated software engineering. Advances in Neural Information Processing Systems **37**, 50528–50652 (2024)
59. Yang, Y., Cheng, W., Chen, S., Zeng, X., Zhang, J., Wang, L., Yu, G., Ma, X., Jiang, Y.G.: Omnisvg: A unified scalable vector graphics generation model. arXiv preprint arXiv:2504.06263 (2025)
60. Yao, S., Chen, H., Yang, J., Narasimhan, K.: Webshop: Towards scalable real-world web interaction with grounded language agents. Advances in Neural Information Processing Systems **35**, 20744–20757 (2022)
61. Ye, Y.: Task memory engine: Spatial memory for robust multi-step llm agents (2025), https://arxiv.org/abs/2505.19436
62. Yi, K., Wu, J., Gan, C., Torralba, A., Kohli, P., Tenenbaum, J.: Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. In: Advances in Neural Information Processing Systems (NeurIPS) 31. pp. 1039–1050 (2018)
63. Zhang, H., Du, W., Shan, J., Zhou, Q., Du, Y., Tenenbaum, J.B., Shu, T., Gan, C.: Building cooperative embodied agents modularly with large language models. arXiv preprint arXiv:2307.02485 (2023)

64. Zhang, Q., Hu, C., Upasani, S., Ma, B., Hong, F., Kamanuru, V., Rainton, J., Wu, C., Ji, M., Li, H., et al.: Agentic context engineering: Evolving contexts for self-improving language models. arXiv preprint arXiv:2510.04618 (2025)
65. Zheng, H., Guan, X., Kong, H., Zheng, J., Zhou, W., Lin, H., Lu, Y., He, B., Han, X., Sun, L.: Pptagent: Generating and evaluating presentations beyond text-to-slides. arXiv preprint arXiv:2501.03936 (2025)
66. Zhou, M., Wang, Y., Hou, J., Zhang, S., Li, Y., Luo, C., Peng, J., Zhang, Z.: Scenex: procedural controllable large-scale scene generation. In: Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence. AAAI'25/IAAI'25/EAAI'25, AAAI Press (2025). https://doi.org/10.1609/aaai.v39i10.33174
67. Zhou, S., Xu, F.F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., et al.: Webarena: A realistic web environment for building autonomous agents. arXiv preprint arXiv:2307.13854 (2023)