HALO: ENABLING SELF-IMPROVING IMAGE GENERATION VIA A HIERARCHICAL AGENTIC LOOP

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world image-generation queries are demanding and often underspecified. They often entail complex specifications, such as spatial layouts and rendered text, while requiring external knowledge that is beyond the generator's internal knowledge. Conventional feed-forward models struggle to handle these queries. Recent methods therefore make image generation agentic by adding reasoning, retrieval and repeated revision. However, they often employ a single loop to plan and revise the whole prompt. This design introduces two key limitations: first, requirements may be omitted or facts lost during planning, prior to image generation; second, correcting a single failed requirement necessitates rewriting the entire prompt, potentially disturbing constraints that have already been satisfied. We introduce HALo, a hierarchical agentic loop with three layers: *layout* specifies what the image should contain, *draft* prepares supporting material for constraints that a textual prompt alone cannot enforce, and *render* generates the image. HALo improves generation at two levels. An inter-layer loop establishes a bidirectional flow. Plans and supporting materials propagate upward from layout to rendering, while unresolved failures are routed downward to the drafting stage for local repair. Crucially, satisfied constraints remain fixed throughout this process, preventing regression in subsequent revisions. Within each layer, an inner loop checks and refines its output in a local context. Intermediate trial-and-error remains local, thus avoiding any rewriting of outputs already established by other layers. Consequently, failed constraints are rectified by updating only their associated supporting material, while the global plan and satisfied constraints remain invariant. Across four benchmarks spanning world knowledge, compositional generation, knowledge-intensive generation and image editing, HALo achieves the best overall performance among the compared methods.

1 INTRODUCTION

Real-world queries for image-generation are inherently characterized by high constraint density and contextual incompleteness. Such queries may simultaneously specify object cardinalities, attribute bindings, spatial relationships and rendered text (Ghosh et al., 2023), while also relying on external knowledge or visual references not available to the generator (Niu et al., 2026; Feng et al., 2026). Conventional feed-forward text-to-image models (Ho et al., 2020; Rombach et al., 2022) produce an output without observing and responding to their own generation, rendering them unable to dynamically adapt to such complex queries. To address this limitation, recent research has reframed image generation as an agentic process (Zhang et al., 2026): iterative closed-loop refinement progressively revises the output across multiple rounds (He et al., 2026b); explicit reasoning unfolds latent user intent (Bian et al., 2026); retrieval-augmented modules supply missing factual or visual context (Shalev-Arkushin et al., 2026; Chen et al., 2026); context compression mitigates semantic drift over extended interactions (Ye et al., 2026b); and code-based tools enforce hard structural and spatial constraints (Ye et al., 2026a).

Despite these advances, agentic generation methods still typically operate within a flat, single-level paradigm: reasoning iteratively refines the generation prompt, the image generator renders it, and the evaluation of the output feeds back into the same reasoning loop (Zhang et al., 2026; Bian et al., 2026; He et al., 2026b). This flat control structure introduces two critical limitations, as illustrated in Figure 1. **First, the absence of a structured target representation undermines global plan-**

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

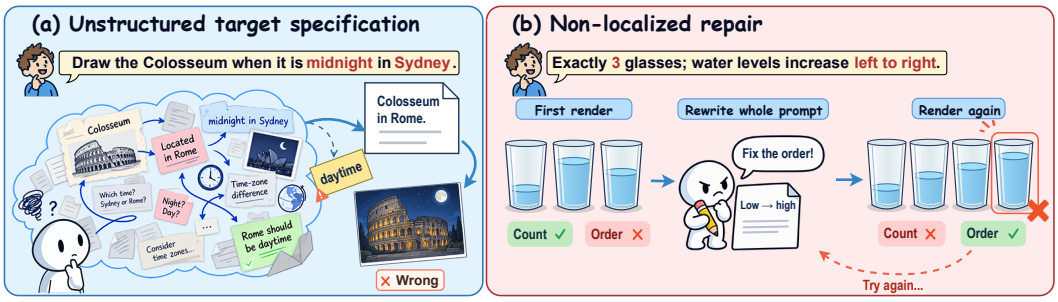


Figure 1: The two limitations of agentic generation methods: (a) requirements may be overlooked or lost without a structured target; (b) rewriting the whole prompt and regenerating the image to fix one constraint may break others that were already satisfied.

ning. Without decomposing the query into explicit requirements, there is no mechanism to ensure comprehensive coverage of all user-specified constraints. Consequently, critical elements may be omitted, while newly retrieved facts or previously determined attributes may be lost during subsequent reasoning steps and thus fail to be transferred accurately to the rendering prompt. **Second, revision remains inefficient because each correction operates at a global granularity rather than on the failed constraint.** Although some methods (Zhang et al., 2026; Ye et al., 2026a) can evaluate the image against a checklist and point to the specific requirement that failed, the correction is still carried out by rewriting the whole prompt and rendering again. These methods therefore cannot confine a correction to the failed constraint: the requirements that already held are resampled along with it, so revision explores broadly instead of repairing locally.

In a word, the core challenge is that *image generation targets comprise heterogeneous constraints, yet existing agentic loops process and refine them at a single granularity*. To bridge this mismatch, we propose the Hierarchical Agentic Loop for image generation (**HALo**), which divides generation into three specialized layers: the *layout* layer specifies what the image should contain, the *draft* layer prepares the supporting material for challenging constraints, and the *render* layer realizes the final image. Its **self-improving** capability operates at two levels: an inter-layer loop that propagates outputs upward and routes unresolved failures downward, and an inner loop that lets each layer verify and refine its own output. **First, the inter-layer loop:** the layout layer passes its plan upward to the draft layer. The draft layer attaches material for constraints that need support and passes the plan with this material to the render layer. If a constraint remains unsatisfied in the rendered image, the failure is routed downward to the draft layer. The draft layer revises the material for that constraint before the image is rendered again. The plan remains unchanged, and constraints that already hold are not rewritten. Each repair therefore updates the material behind a failed constraint rather than rewriting the whole prompt. **Second, the inner loop:** each layer operates in a layer-specific context and may take several turns and call the tools it needs. It iteratively improves its output through self-checking. The *layout* layer constructs a structured plan of explicit constraints and revises it until the plan is internally consistent and covers the query. The *draft* layer grounds constraints that are difficult to satisfy from the instruction or parametric knowledge alone, and rebuilds material that fails its own check. The *render* layer generates an image, checks it against the constraints, and retries failed constraints. Intermediate trials and corrections remain local until the layer's output is passed upward. Local trial-and-error therefore does not rewrite outputs established by other layers. Extensive experiments across four benchmarks demonstrate that HALo consistently outperforms traditional image-generation models and agentic methods built on the same image backbone.

Our contributions are threefold. **(i)** We identify two limitations of single-level agentic generation: (a) entangled heterogeneous constraints hinder accurate global planning, and (b) revision relies on global prompt rewriting, leaving corrections insufficiently localized. **(ii)** We propose HALo, which organizes generation into layout, draft and render layers: an inner loop lets each layer refine its output, and an inter-layer loop sends unresolved failures down to the draft layer after render retries. **(iii)** We evaluate HALo across four benchmarks spanning world knowledge, compositional, knowledge-intensive and image-editing generation, demonstrating consistent gains and validating the design rationale through layer-specific behavioral analysis.

2 Related Work

2.1 Image Generation Models

Diffusion models established the foundation of modern image generation by learning to reverse a gradual noising process (Ho et al., 2020). Latent diffusion subsequently moved this process into a compressed representation and introduced text conditioning through cross-attention (Rombach et al., 2022). Replacing convolutional backbones with transformers over latent patches further improved the scalability of image generation (Peebles & Xie, 2023). Control branches enabled direct conditioning on structural signals such as edges, depth and pose (Zhang et al., 2023), while advances in data construction and training improved text rendering and image editing within the same generative framework (Wu et al., 2025). A parallel line of work integrates visual understanding and generation, either by combining autoregressive and diffusion decoding for tasks such as visual question answering, text-to-image generation and inpainting (Xie et al., 2025), or by scaling interleaved multimodal pretraining to support understanding, generation and editing in a unified model (Deng et al., 2025). These advances strengthen the rendering model itself, whereas agentic methods focus on how reasoning, external tools and feedback are organized around it.

2.2 Agentic Image Generation

Agentic image generation has explored several complementary capabilities. Planning-oriented methods represent visual requirements before rendering, for example by converting text into object layouts (Feng et al., 2023), organizing context across planning levels (Zhang et al., 2026), or using executable sketches to ground layout, count and text (Ye et al., 2026a). Other methods enrich the generation context through iterative self-refinement (Yang et al., 2024), reference retrieval (Shalev-Arkushin et al., 2026), search and reasoning (He et al., 2026a; Bian et al., 2026), or learned search policies (Chen et al., 2026; Feng et al., 2026). Recent systems also incorporate memory and visual feedback to support longer generation trajectories (He et al., 2026b; Zhang et al., 2026).

Together, these works establish the main ingredients of agentic image generation, but differ in how they represent, ground and repair the generation target across stages. Among them, GenClaw (Ye et al., 2026a) is particularly close to our setting, using code as an executable visual canvas between conceptualization and rendering. HALo instead introduces a structured, entry-level target that remains invariant across planning, grounding, rendering and repair, while treating sketches and retrieved references as supporting material attached to individual target entries. This shared representation provides feedback with a concrete unit of repair: a failed requirement can be revisited while previously satisfied requirements remain fixed in the target.

3 Method

3.1 Overview

Task formulation. Let q denote a query consisting of a natural-language instruction and optional source or reference images. A generation system $\mathcal{G}$ aims to produce a consistent image I satisfying the query:

$$I = \mathcal{G}(q). \tag{1}$$

The query may specify objects, attributes, spatial relations, rendered text, or image edits. For editing, I should incorporate requested changes while preserving source content unrelated to those changes.

Framework. HALo organizes generation into layers: layout, draft and render. The layout layer determines what the image should contain, the draft layer prepares supporting material for challenging constraints, and the render layer generates the image. As illustrated in Figure 2, self-improvement operates at two levels. First, an inter-layer loop passes the plan and supporting material upward from layout to render, while routing unresolved render failures downward to the draft layer. The plan remains fixed during this feedback. Thus, repair changes the material behind a failed constraint rather than rewriting the target. Second, an inner loop lets each layer check and refine its output within a layer-specific context. Intermediate trials remain local to the responsible layer until its output is passed upward. Sections 3.2 - 3.4 detail the two loops and how each layer can be extended.

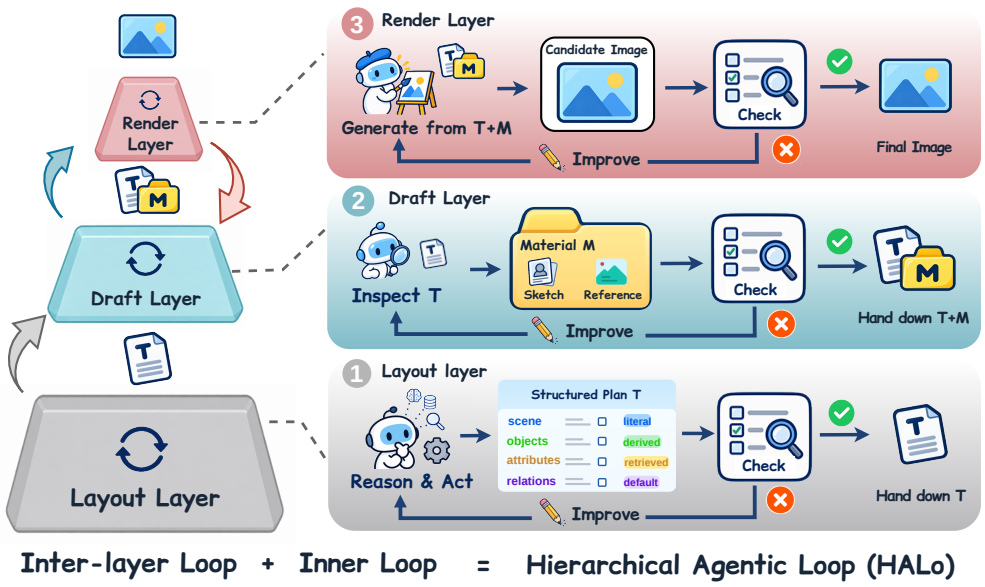


Figure 2: Framework of HALo: the layout, draft and render layers each close their own inner loop, and an inter-layer loop passes checked results up or sends an unresolved failure down.

3.2 Inter-Layer Loop

Let $\mathcal{L}$, $\mathcal{D}$ and $\mathcal{R}$ denote the layout, draft and render operations, respectively. Their outputs are a structured plan T of target constraints, supporting material M and an image I.

Verified forward propagation. Let $V_{\mathcal{L}}(T)$ and $V_{\mathcal{D}}(T, M)$ denote Boolean results of the layout and draft checks, respectively. These checks are detailed in Section 3.3. The outputs propagate upward from layout through draft to render as follows:

$$\begin{cases} T = \mathcal{L}(q), & \text{given } q, \\ M = \mathcal{D}(T), & \text{if } V_{\mathcal{L}}(T) = \text{true}, \\ I = \mathcal{R}(T, M), & \text{if } V_{\mathcal{D}}(T, M) = \text{true}. \end{cases} \tag{2}$$

After layout validation, the plan T remains fixed throughout generation and repair. The render layer treats both T and M as read-only inputs.

Failure-driven feedback. Let $\mathcal{F}$ denote the set of entries in T that fail the checks on the current image I. If $\mathcal{F} \neq \varnothing$ after render refinement, these entries and I are returned to the draft layer. This feedback reveals gaps in material selection or construction. The draft layer then revises or constructs material for the failed entries. After draft validation, it retains material for passing entries and replaces material for failed entries:

$$\begin{aligned} M_{\overline{\mathcal{F}}} &= \{\, m_e \in M \mid e \notin \mathcal{F} \,\}, \\ M^{+}_{\mathcal{F}} &= \{\, m^{+}_e \mid e \in \mathcal{F} \,\}, \\ M^{+} &= M_{\overline{\mathcal{F}}} \cup M^{+}_{\mathcal{F}}, \\ I^{+} &= \mathcal{R}(T, M^{+}). \end{aligned} \tag{3}$$

The render layer checks I^{+} against T and M^{+}, and updates $\mathcal{F}$. Feedback does not reopen the layout layer. The plan T is derived from the original query and checked for consistency and coverage. It remains fixed so that repair changes the supporting material without changing the requirements.

3.3 Inner Loop

Each layer is an iterative loop rather than a single step. Within its own context, it may take several turns and call the tools it needs. Self-checking guides these revisions toward an output that passes the

layer's checks. Intermediate trials remain local, while outputs and feedback are exchanged through the inter-layer loop. This section details the layer operations and checks introduced in Section 3.2.

Layout layer: producing the plan. The layout layer $\mathcal{L}$ examines the query q to identify explicit requirements. It infers implicit constraints through reasoning and retrieves external facts when needed, using the retrieved information to refine its conclusions. It organizes the resulting requirements into four arrays with distinct roles:

$$T = (\text{scene}, \text{objects}, \text{attributes}, \text{relations}). \tag{4}$$

The *scene* array records global context such as location, lighting and time of day. The *objects* array names each entity and marks whether it is central to the request. The *attributes* array associates each object with properties such as color or pose. The *relations* array specifies ordering and spatial relationships among objects. For entry-level operations, we also use T to denote the collection of entries across these arrays. Each entry e of T is a constraint the image must satisfy and carries a provenance label indicating its origin:

$$e.\text{source} \in \{\textit{literal}, \textit{derived}, \textit{retrieved}, \textit{default}\}, \tag{5}$$

where *literal* denotes a value stated in the query, *derived* a value inferred through reasoning, *retrieved* a value obtained through search, and *default* a value assigned when unspecified. A structural check verifies that T follows the schema. A semantic check verifies that T is internally consistent and covers all requirements in the query. The layer revises and rechecks the plan if either check fails. The loop terminates when both checks pass, yielding $V_{\mathcal{L}}(T) = \text{true}$.

Each derived entry includes two metadata fields. The *basis* field records the evidence or reasoning supporting its inferred value, while *instead_of* records the interpretation ruled out by that reasoning. Consider the query *"Draw the Colosseum when it is midnight in Sydney."* The time-zone difference implies daytime in Rome, but Figure 1(a) illustrates how this conclusion can be lost before rendering, resulting in a nighttime Colosseum. The entry records *source* = derived, *value* = daytime, *basis* = the time-zone offset between the two cities, and *instead_of* = a nighttime Colosseum. These fields explain why a daytime Colosseum is consistent with midnight in Sydney. They allow the semantic check to assess the time-zone inference rather than overwrite it with a nighttime interpretation. Appendix details the fields and required metadata for each entry type, the four provenance labels and the *basis* and *instead_of* fields for derived entries, and the validation criteria for schema compliance, internal consistency and query coverage.

Draft layer: producing the material. The draft layer $\mathcal{D}$ selects a subset $T_{\text{sup}} \subseteq T$ whose constraints are unlikely to be satisfied by the instruction alone. These include exact counts, orderings and spatial relations, as well as requirements that depend on external visual references. For each selected entry e, it produces material m_e as a code-rendered sketch or a retrieved reference:

$$M = \{m_e \mid e \in T_{\text{sup}}\}, \tag{6}$$

with $M = \varnothing$ when no entry requires support. The layer checks each m_e against its interpretation of e to verify the intended structure, fact or appearance. Any item that fails is revised or replaced and checked again. The validation result $V_{\mathcal{D}}(T, M)$ is true when all material passes these checks.

Render layer: producing the image. The render layer $\mathcal{R}$ produces an image I from the plan T and material M. It checks each entry $e \in T$ using M as supporting evidence. The Boolean function $\text{check}(e, I, M)$ returns true if e is judged satisfied and false otherwise. The failure set introduced in Section 3.2 is therefore

$$\mathcal{F} = \{\, e \in T \mid \neg\, \text{check}(e, I, M) \,\}. \tag{7}$$

If $\mathcal{F}$ is nonempty, the layer records one correction for each failed entry. It then retries rendering once while retaining already satisfied requirements. Afterward, it checks the new image and recomputes $\mathcal{F}$. The inner loop terminates when $\mathcal{F}$ is empty or after this retry.

3.4 Extensibility

The layers communicate through the plan T, supporting material M, and render feedback comprising the image I and failure set $\mathcal{F}$. Each layer can be extended independently as long as these interfaces are preserved. The layout layer can incorporate domain-specific knowledge within the existing plan schema. The draft layer can add sketch renderers or retrieval sources. The render layer can adopt alternative image models or checkers while retaining entry-level feedback. Each layer continues to validate its own output after an extension.

Table 1: Results on WISE and GenEval++ (%). WISE: Cult.=culture; Time/Space=temporal/spatial reasoning; Bio/Phys/Chem=biology/physics/chemistry. GenEval++: C/Cnt/P=color/count/position; combined labels denote joint constraints; M-Cnt=multi-object counting. Overall is weighted for WISE and averaged for GenEval++. †: WISE results from He et al. (2026a); BAGEL uses CoT on WISE; "–": unavailable. Best in **bold**, second best underlined.

	WISE							GenEval++							
Method	Cult.	Time	Space	Bio	Phys	Chem	Overall	Count	Color	C/Cnt	C/Pos	P/Cnt	P/Size	M-Cnt	Overall
Direct generation															
FLUX.1-dev† (Black Forest Labs, 2024)	48.0	58.0	62.0	42.0	51.0	35.0	49.0	60.0	40.0	25.0	25.0	7.5	40.0	30.0	32.5
SD-3.5-Medium† (Esser et al., 2024)	43.0	50.0	52.0	41.0	53.0	33.0	44.7	60.0	40.0	25.0	25.0	7.5	40.0	30.0	32.5
GPT-Image-1† (OpenAI, 2025b)	81.0	71.0	89.0	83.0	79.0	74.0	79.9	67.5	90.0	72.5	62.5	60.0	80.0	85.0	73.9
BAGEL (Deng et al., 2025)	78.0	63.3	56.7	37.5	55.0	50.8	62.8	60.0	32.5	25.0	32.5	25.0	47.5	37.5	37.1
Qwen-Image† (Wu et al., 2025)	62.0	63.0	77.0	57.0	75.0	40.0	62.2	72.5	87.5	72.5	60.0	47.5	72.5	55.0	66.8
Gemini-2.5-Flash-Image (Google, 2025)	–	–	–	–	–	–	–	70.0	80.0	68.0	53.0	68.0	78.0	80.0	71.0
DraCo (Jiang et al., 2026a)	–	–	–	–	–	–	–	53.0	45.0	28.0	40.0	28.0	53.0	38.0	40.7
GPT-Image-1.5 (OpenAI, 2025c)	89.0	69.2	88.3	80.0	75.8	77.5	82.5	**95.0**	85.0	77.5	50.0	60.0	77.5	80.0	75.0
Gemini-3.1-Flash-Image (Google DeepMind, 2026)	87.5	82.5	91.7	86.7	82.5	86.7	86.6	85.0	85.0	80.0	75.0	67.5	72.5	77.5	77.5
Qwen-Image-2.0 (Zhao et al., 2026)	82.2	65.0	89.9	79.2	80.0	74.8	79.5	–	–	–	–	–	–	–	–
SenseNova-U1-A3B-MoT (w/ CoT) (Diao et al., 2026)	83.3	77.5	91.7	84.2	80.0	84.2	83.4	–	–	–	–	–	–	–	–
Agentic generation															
Gemini-3-Pro-Image (Google DeepMind, 2025)	89.8	81.7	**93.3**	81.7	86.7	87.5	87.6	90.0	80.0	75.0	70.0	72.5	72.5	85.0	77.9
GenAgent† (Jiang et al., 2026b)	78.0	67.0	78.0	72.0	77.0	55.0	73.1	77.5	77.5	65.0	80.0	60.0	72.5	75.0	72.5
Mind-Brush† (He et al., 2026a)	83.0	69.0	84.0	71.0	85.0	68.0	78.4	70.0	77.5	77.5	75.0	85.0	77.5	85.0	78.2
GenClaw (Ye et al., 2026a)	**94.5**	82.5	91.7	83.3	84.2	87.5	89.3	**95.0**	87.5	82.5	**85.0**	**92.5**	82.5	90.0	87.9
Qwen-Image-Agent (Zhang et al., 2026)	92.0	91.7	**93.3**	83.3	86.7	90.0	90.2	–	–	–	–	–	–	–	–
HALo	**94.5**	**99.2**	92.5	**88.3**	**91.7**	**94.2**	**93.7**	92.5	**95.0**	**87.5**	**85.0**	87.5	**87.5**	**92.5**	**89.6**

4 EXPERIMENTS

4.1 SETUP

Benchmarks. We evaluate HALo on four benchmarks covering world knowledge, compositional generation, knowledge-intensive generation and image editing. WISE (Niu et al., 2026) evaluates world knowledge through prompts spanning cultural, spatio-temporal and natural-science domains, with per-category correctness rates and a weighted overall score. GenEval++ (Ye et al., 2025) evaluates compositional instruction following through prompts combining object counts, colors, spatial positions and relative sizes. Mind-Bench (He et al., 2026a) evaluates knowledge-intensive generation, counting an image as successful only when it satisfies every requirement in its checklist. RISEBench (Zhao et al., 2025) evaluates image editing from a source image and an instruction, assessing appearance consistency, instruction reasoning and visual plausibility.

Baselines. We compare HALo with two groups: (1) direct generation models, including open-source text-to-image and unified models (Qwen-Image, FLUX, SD-3.5 and BAGEL) and proprietary models (GPT-Image and Gemini); and (2) agentic generation systems incorporating reasoning, retrieval or revision (Gemini-3-Pro-Image, GenAgent, Mind-Brush, GenClaw and Qwen-Image-Agent).

Implementation. We use GPT-5.5 (OpenAI, 2025a) as the agent backbone and Gemini-3.1-Flash-Image (Google DeepMind, 2026) as the image model across all four benchmarks. The evaluation models are Qwen3.5-35B for WISE, GPT-4.1 for GenEval++ and RISEBench, and Gemini-3-Pro for Mind-Bench. We use the metric definitions provided by each benchmark.

4.2 MAIN RESULTS

4.2.1 AN EXPLICIT PLAN PRESERVES REASONING CONCLUSIONS

A valid reasoning outcome is only valuable if it is faithfully realized in the final rendering. On WISE, HALo achieves an overall score of 93.7, significantly outperforming direct generation baselines using the same backbone (86.6, Table 1). This gap is most pronounced in the Time category, where performance rises from 82.5 to 99.2. While other methods may derive correct reasoning outcomes, these insights often remain implicit and are susceptible to being overwritten by the generator's priors or the original query's ambiguity. In contrast, HALo explicitly encodes semantic requirements as immutable constraints within the target representation. Once validated for consistency at the layout

stage, this plan is frozen and propagated downstream, ensuring that rendering strictly adheres to the verified logic rather than reconstructing it from scratch.

Such structured planning supports domain-specific reasoning. In Phys, the layout layer translates physical laws into explicit constraints in T, and HALo achieves a leading score of 91.7. These constraints are recorded in the entries without changing the interfaces to the draft and render layers. On Space, HALo performs comparably to existing methods, as spatial constraints often depend more on structural guidance than on complex inference. GenClaw provides such guidance through code-based scene construction. Rather than introducing tools, HALo organizes planning, supporting material and feedback into a hierarchical loop. Overall, these results support the value of explicit planning for knowledge-dependent generation, with benefits that vary across requirement types.

Table 2: Mind-Bench success rates, requiring all checklist items to pass. SE: special events; MC: movie characters; IP: intellectual-property-related objects; WK: world knowledge; SL: science and logic; LR: life reasoning; GU: geographic understanding. Overall averages category scores; "–" denotes unsupported image-input tasks. Best in **bold**, second best underlined.

	Knowledge-Driven					Reasoning-Driven					
Model Name	SE	Weather	MC	IP	WK	SL	Poem	LR	GU	Math	Overall
Direct generation											
FLUX.1-dev (Black Forest Labs, 2024)	0.04	0.00	0.00	0.00	0.02	0.02	0.04	–	–	–	0.02
BAGEL (Deng et al., 2025)	0.02	0.00	0.00	0.00	0.00	0.02	0.02	0.02	0.00	0.08	0.02
Qwen-Image (Wu et al., 2025)	0.08	0.00	0.04	0.00	0.00	0.04	0.00	0.04	0.00	0.00	0.02
GPT-Image-1 (OpenAI, 2025b)	0.32	0.06	0.22	0.02	0.16	0.32	0.10	0.24	0.10	0.12	0.17
GPT-Image-1.5 (OpenAI, 2025c)	0.36	0.18	0.22	0.04	0.30	0.34	0.08	0.34	0.10	0.02	0.20
Gemini-2.5-Flash-Image (Google, 2025)	0.24	0.20	0.12	0.00	0.36	0.32	0.40	0.28	0.08	0.24	0.22
FLUX.2-Pro (Black Forest Labs, 2025)	0.28	0.32	0.02	0.00	0.20	0.36	0.58	0.20	0.16	0.12	0.22
Agentic generation											
Gemini-3-Pro-Image (Google DeepMind, 2025)	0.52	0.24	0.20	0.04	0.52	0.60	0.72	0.28	**0.44**	0.28	0.38
Mind-Brush (He et al., 2026a)	0.54	0.16	0.62	0.18	0.40	0.26	0.54	0.10	0.16	0.14	0.31
GenClaw (Ye et al., 2026a)	0.64	**0.44**	0.66	**0.32**	0.64	**0.78**	0.90	**0.38**	0.32	0.60	0.57
HALo	**0.66**	0.42	**0.82**	0.26	**0.72**	0.70	**0.98**	0.34	0.34	**0.72**	**0.60**

4.2.2 Requirement-aligned support for knowledge-intensive generation

Mind-Bench evaluates a diverse spectrum of capabilities, ranging from factual identification to reference-sensitive generation and visual reasoning. HALo achieves an overall score of 0.60, outperforming the strong baseline GenClaw (0.57, Table 2). The key challenge in this benchmark lies not merely in retrieving missing information, but in maintaining strict alignment between retrieved evidence and the specific requirements it addresses. Unlike prompt-level retrieval methods, HALo attaches each retrieved fact or visual reference directly to its corresponding entry in the target representation. This design ensures that supporting information remains semantically bound to its target throughout rendering and verification. Case studies on Mind-Bench illustrate the effectiveness of this alignment mechanism. In logo composition and poem visualization, retrieval identifies factual elements (e.g., corporate marks, titles), while entries in T enforce structural constraints (e.g., spatial ordering, metaphorical interpretation). In both scenarios, retrieved context is effective precisely because it is tied to explicit requirements rather than appended as undifferentiated background noise. Consequently, the render layer can synthesize images using both visual evidence and structural constraints from T, enabling precise verification against the same entries. For tasks not requiring external search, the primary bottleneck shifts to decomposing multi-step deliverables and verifying stage-wise realization. Here, the explicit structure of T concretizes abstract stages, allowing local feedback loops to pinpoint unmet requirements. Ultimately, these results demonstrate that the hierarchical loop's value stems from organizing retrieval, planning, and feedback around shared requirements, rather than from retrieval capability alone.

4.2.3 Separating edit and preservation requirements

RISEBench demands precise trade-offs between applying edits and preserving source content. HALo matches GenClaw in instruction reasoning (74.8 vs. 74.3) while surpassing it in appearance

consistency (93.8 vs. 88.1) and strict accuracy (54.7 vs. 46.7), with comparable visual plausibility (Table 3). This suggests that the key challenge lies in executing edits without perturbing preserved regions. HALo addresses this by encoding both modification and preservation requirements as explicit entries in T, enabling strategic integration of changes with source content. Supporting materials are selectively attached to relevant entries, and the render layer synthesizes outputs using the source image and T. Upon failure, feedback diagnoses whether the issue concerns the edit or preservation, triggering targeted repair from the fixed plan rather than global prompt rewriting. This prevents error accumulation and protects satisfied constraints.

Table 3: RISEBench results (%). Instr. Reas.: instruction reasoning; App. Consis.: appearance consistency with the source image; Vis. Plaus.: visual plausibility. Acc. reports percentage of cases scoring full marks across all dimensions. BAGEL uses CoT. Best in **bold**, second best underlined.

Model Name	Instr. Reas.	App. Consis.	Vis. Plaus.	Acc.
Direct generation				
FLUX.1-dev (Black Forest Labs, 2024)	26.0	71.6	85.2	1.9
GPT-Image-1 (OpenAI, 2025b)	62.8	80.2	94.9	28.9
BAGEL (Deng et al., 2025)	45.9	73.8	80.1	11.9
Qwen-Image (Wu et al., 2025)	49.9	71.0	91.5	19.4
Gemini-2.5-Flash-Image (Google, 2025)	61.2	86.0	91.3	32.8
Gemini-3.1-Flash-Image (Google DeepMind, 2026)	72.9	84.6	93.2	46.9
Agentic generation				
Gemini-3-Pro-Image (Google DeepMind, 2025)	**77.0**	85.5	94.4	47.2
Mind-Brush (He et al., 2026a)	61.5	79.4	86.5	24.7
GenClaw (Ye et al., 2026a)	74.3	88.1	**95.9**	46.7
HALo	74.8	**93.8**	95.5	**54.7**

4.3 Analysis of Inter-Layer Loop

We analyze generation traces to characterize the effectiveness of HALo's internal feedback loop. A case triggers the repair process only when the initial review flags at least one entry in T as `NEEDS_FIX`. Crucially, the system prioritizes local rendering retries; a case is routed back to the draft layer solely when the review indicates that existing support is insufficient. We distinguish local render retries from draft-layer returns to expose where each failure is addressed. Throughout this process, the plan T remains frozen, and pre-existing materials are preserved. Table 4 summarizes these dynamics. The initial internal review pass rate varies across benchmarks, ranging from 95.24% on WISE to 62.97% on RISEBench, reflecting how often the first rendered image satisfies the checked requirements. Among initially failed cases with a completed final review, 71.79% to 91.30% pass after repair. These recovery rates show that targeted correction can resolve many failures while keeping the plan T fixed.

Draft-return requests range from 0.00% to 8.93% of initially failed cases. On GenEval++, no draft-return requests occur, although the draft layer is still used in the forward pass. Code-rendered sketches provide structural support for most exact-count and positioning requirements, while failures

Table 4: Internal review and repair behavior, in percent. **Initial review pass**: cases passing the first internal review; **Review recovery**: initially failed cases with a completed final review that pass after repair; **Draft-return requests**: initially failed cases requesting draft revision. The first two metrics use cases with a completed final review, whereas draft-return requests use all initially failed cases.

Benchmark	Initial review pass	Review recovery	Draft-return requests
WISE	95.24	91.30	8.93
GenEval++	87.05	77.78	0.00
Mind-Bench	74.90	78.05	3.20
RISEBench	62.97	71.79	6.82

remaining after render retries are diagnosed as rendering noncompliance. This pattern highlights a conditional division of labor: render retries address failures in following existing support, while draft returns address insufficient supporting material. The contribution of the inter-layer loop to benchmark performance is evaluated separately in the ablation study. These metrics describe internal review behavior and should not be interpreted as benchmark scores.

4.4 Ablation Study

We conduct an ablation study on a fixed subset of 100 WISE cases to isolate the contributions of the two self-improvement mechanisms in HALo. All variants share an identical experimental setup. As detailed in Table 5, the single-pass hierarchy without either loop achieves a score of 86. Enabling only the inner loop improves performance to 93, demonstrating the effectiveness of local self-correction within individual layers. Furthermore, incorporating the inter-layer loop yields a final score of 97. This incremental gain of 4 points over the inner-loop-only variant shows that the inter-layer loop provides complementary benefits beyond local refinement. It returns unresolved render failures to the draft layer for targeted revision of their supporting material.

Table 5: Ablation on 100 WISE cases. All variants share the same setup.

Variant	Inner loop	Inter-layer loop	WISE score
Single-pass hierarchy			86
Inner loop only	✓		93
Full HALo	✓	✓	**97**

4.5 Case Study

Figure 3 traces an exact-count constraint through four stages of the loop. The checked plan fixes c_2 as "exactly eight legs," while the draft layer initially supplies no extra material. The first image and a render-layer retry both fail, indicating insufficient supporting material rather than an incorrect target. The draft layer then constructs a checked structural sketch for c_2, enabling the render layer to produce a correct image without reopening or modifying T. This case illustrates how inter-layer feedback localizes repair to the material behind an unresolved constraint while preserving the checked target. Additional case analyses are provided in the appendix.

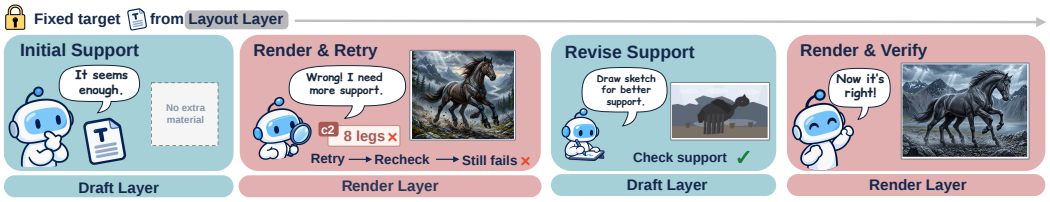


Figure 3: Representative repair trajectory for an exact-count constraint. The checked plan specifies an eight-legged horse and remains fixed. The first image has six legs, and the render retry fails. The draft layer supplies a checked structural sketch for the render layer to generate and verify the image.

5 Conclusion

We introduced HALo, a hierarchical agentic loop that enables self-improving image generation under heterogeneous constraints. The layout, draft and render layers each close their own check-and-improve loop, while cross-layer feedback escalates unresolved failures to the layer that can address them without reopening verified requirements. These results show that self-checking and targeted cross-layer repair improve generation without disturbing satisfied requirements. Across four benchmarks spanning world knowledge, compositional generation, knowledge-intensive generation and image editing, HALo achieves the strongest aggregate performance among the compared methods, demonstrating the benefits of self-improvement both within and across layers.

AI USE STATEMENT

We used AI tools to polish the manuscript and provide limited assistance with coding and running experiments. The authors take full responsibility for the paper's content, experimental results, and conclusions.

REFERENCES

Feifei Bian, Zhimin Zheng, Wei Deng, Daiguo Zhou, and Jian Luan. Rs-gen: A multi-stage agentic framework for reasoning and search-augmented image generation. *arXiv preprint arXiv:2606.23221*, 2026.

Black Forest Labs. FLUX. https://github.com/black-forest-labs/flux, 2024.

Black Forest Labs. FLUX.2: Frontier visual intelligence. https://bfl.ai/blog/flux-2, 2025.

Shuang Chen, Quanxin Shou, Hangting Chen, Yucheng Zhou, Kaituo Feng, Wenbo Hu, Yi-Fan Zhang, Yunlong Lin, Wenxuan Huang, Mingyang Song, et al. Unify-agent: A unified multimodal agent for world-grounded image synthesis. *arXiv preprint arXiv:2603.29620*, 2026.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

Haiwen Diao, Penghao Wu, Hanming Deng, Jiahao Wang, Shihao Bai, Silei Wu, Weichen Fan, Wenjie Ye, Wenwen Tong, Xiangyu Fan, et al. Sensenova-u1: Unifying multimodal understanding and generation with neo-unify architecture. *arXiv preprint arXiv:2605.12500*, 2026.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, 2024. arXiv:2403.03206.

Kaituo Feng, Manyuan Zhang, Shuang Chen, Yunlong Lin, Kaixuan Fan, Yilei Jiang, Hongyu Li, Dian Zheng, Chenyang Wang, and Xiangyu Yue. Gen-searcher: Reinforcing agentic search for image generation. *arXiv preprint arXiv:2603.28767*, 2026.

Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in neural information processing systems*, 36: 18225–18250, 2023.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36: 52132–52152, 2023.

Google. Introducing gemini 2.5 flash image (nano banana), our state-of-the-art image model. https://developers.googleblog.com/en/introducing-gemini-2-5-flash-image/, 2025.

Google DeepMind. Nano banana pro: Our gemini 3 pro image model. https://blog.google/technology/ai/nano-banana-pro/, 2025.

Google DeepMind. Gemini 3.1 Flash Image model card. https://deepmind.google/models/model-cards/gemini-3-1-flash-image, 2026.

Jun He, Junyan Ye, Zilong Huang, Dongzhi Jiang, Chenjue Zhang, Leqi Zhu, Renrui Zhang, Xiang Zhang, and Weijia Li. Mind-brush: Integrating agentic cognitive search and reasoning into image generation. *arXiv preprint arXiv:2602.01756*, 2026a.

Zefeng He, Siyuan Huang, Xiaoye Qu, Yafu Li, Tong Zhu, Yu Cheng, and Yang Yang. Gems: Agent-native multimodal generation with memory and skills. *arXiv preprint arXiv:2603.28088*, 2026b.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Dongzhi Jiang, Renrui Zhang, Haodong Li, Zhuofan Zong, Ziyu Guo, Jun He, Claire Guo, Junyan Ye, Rongyao Fang, Weijia Li, Rui Liu, and Hongsheng Li. Draco: Draft as cot for text-to-image preview and rare concept generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026a. arXiv:2512.05112.

Kaixun Jiang, Yuzheng Wang, Junjie Zhou, Pandeng Li, Zhihang Liu, Chen-Wei Xie, Zhaoyu Chen, Yun Zheng, and Wenqiang Zhang. Genagent: Scaling text-to-image generation via agentic multimodal reasoning. *arXiv preprint arXiv:2601.18543*, 2026b.

Yuwei Niu, Munan Ning, Mengren Zheng, Weiyang Jin, Bin Lin, Peng Jin, Jiaqi Liao, Chaoran Feng, Fanqing Meng, Kunpeng Ning, Bin Zhu, and Li Yuan. Wise: A world knowledge-informed semantic evaluation for text-to-image generation. In *International Conference on Machine Learning*, 2026.

OpenAI. OpenAI GPT-5 System Card. *arXiv preprint arXiv:2601.03267*, 2025a.

OpenAI. Introducing our latest image generation model in the api (gpt-image-1). `https://openai.com/index/image-generation-api/`, 2025b.

OpenAI. GPT-Image-1.5. OpenAI model documentation, `https://platform.openai.com/docs/models`, 2025c.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Rotem Shalev-Arkushin, Rinon Gal, Amit Bermano, and Ohad Fried. Imagerag: Dynamic image retrieval for reference-guided image generation. In *International Conference on Learning Representations*, volume 2026, pp. 11400–11427, 2026.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. In *International Conference on Learning Representations*, volume 2025, pp. 28240–28264, 2025.

Zhengyuan Yang, Jianfeng Wang, Linjie Li, Kevin Lin, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. Idea2img: Iterative self-refinement with gpt-4v for automatic image design and generation. In *European conference on computer vision*, pp. 167–184. Springer, 2024.

Junyan Ye, Dongzhi Jiang, Zihao Wang, Leqi Zhu, Zhenghao Hu, Zilong Huang, Jun He, Zhiyuan Yan, Jinghua Yu, Hongsheng Li, et al. Echo-4o: Harnessing the power of gpt-4o synthetic images for improved image generation. *arXiv preprint arXiv:2508.09987*, 2025.

Junyan Ye, Jun He, Zilong Huang, Dongzhi Jiang, Xuan Yang, Rui Chen, and Weijia Li. Genclaw: Code-driven agentic image generation. *arXiv preprint arXiv:2605.30248*, 2026a.

Yuxiao Ye, Haoran He, Fangyuan Kong, Xintao Wang, Pengfei Wan, Kun Gai, and Ling Pan. Edit-r2: Context-aware reinforcement learning for multi-turn image editing. *arXiv preprint arXiv:2606.05950*, 2026b.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3813–3824. IEEE, 2023.

Zekai Zhang, Jiahao Li, Jie Zhang, Kaiyuan Gao, Kun Yan, Lihan Jiang, Ningyuan Tang, Shengming Yin, Tianhe Wu, Xiaoyue Chen, et al. Qwen-image-agent: Bridging the context gap in real-world image generation. *arXiv preprint arXiv:2606.26907*, 2026.

Bing Zhao, Chenfei Wu, Deqing Li, Hao Meng, Jiahao Li, Jie Zhang, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kuan Cao, Kun Yan, Liang Peng, Lihan Jiang, Niantong Li, et al. Qwen-image-2.0 technical report. *arXiv preprint arXiv:2605.10730*, 2026.

Xiangyu Zhao, Peiyuan Zhang, Kexian Tang, Xiaorong Zhu, Hao Li, Wenhao Chai, Zicheng Zhang, Renqiu Xia, Guangtao Zhai, Junchi Yan, Hua Yang, Xue Yang, and Haodong Duan. Envisioning beyond the pixels: Benchmarking reasoning-informed visual editing. In *Advances in Neural Information Processing Systems*, 2025.