EvoDesign: Agentic Editable Diagram Creation via Design Expertise Evolution

Tianfu Wang[1,*,‡] **Leilei Ding**[2,*] **Ziyang Tao**[2,*] **Yi Zhan**[2,*] **Zhiyuan Ma**[2] **Wei Wu**[2] **Yuxuan Lei**[2] **Junyang Wang**[2] **Yizhao Xu**[3] **Yin WU**[1] **Zhengyu Hu**[1] **Hongyuan Zhu**[2] **Nicholas Jing Yuan**[1] **Yanyong Zhang**[2] **Hui Xiong**[1,4,†]

[1]Hong Kong University of Science and Technology (Guangzhou)
[2]University of Science and Technology of China [3]Peking University
[4]Hong Kong University of Science and Technology

[*]Equal contribution [‡]Project Leader [†]Corresponding author
[‡]tianfuwang.cs@gmail.com [†]xionghui@ust.hk

Abstract

High-fidelity diagram creation requires coordinated visual-spatial decisions over semantic topology, visual styling, and geometric layout. Existing methods face a representation gap: pixel-based generation offers limited object-level control, while code-based synthesis provides executable structure at the cost of intuitive manipulation. We introduce EvoDesign, an agentic framework for editable diagram creation through an intermediate canvas schema that is both machine-actionable and directly manipulable by users. EvoDesign first constructs a diagram manifest through coordinated structure, style, and layout agents, and then renders the manifest in a diagnostic verification environment. This environment produces objective defect signals and VLM-based critique, enabling localized refinement of the current diagram while also providing evidence for long-term design expertise evolution. The evolution mechanism distills refinement traces into a three-tier design expertise memory, where candidate strategies are linked to the verification evidence that supports them and are promoted into broader guidelines and principles only through repeated cross-context support. We further define CanvasBench, a canvas-recoverable benchmark and evaluation protocol with content, visual, and cognitive dimensions. Experiments across pixel, code-based, agentic, one-shot schema, and canvas-native methods show that EvoDesign provides the strongest overall balance across content fidelity, visual organization, and cognitive usability. Our code is available at https://github.com/GeminiLight/EvoDesign.

1 Introduction

Diagrams such as flowcharts, concept maps, and system architecture graphs serve as ubiquitous visual artifacts for organizing and communicating complex information [17]. By transforming abstract entities and relations into spatial structures, diagrams reduce cognitive load, accelerate comprehension, and support collaborative reasoning. High-fidelity diagram creation, however, requires more than producing a visually plausible image. It requires coordinated decisions over semantic topology, visual styling, and geometric layout, where local changes in one layer can alter the communicative meaning of the whole diagram [13].

Existing automated diagramming methods generally rely on either pixel-level generation or executable code synthesis. Pixel-based generative models can produce visually rich diagram-like renderings, but they provide weak object-level control and may introduce semantic hallucinations, illegible text, or unstable relations [36, 2, 7]. Code-based methods, including LaTeX, TikZ, Mermaid, and SVG-oriented synthesis, improve structural executability through symbolic programs [34, 25]. Yet their outputs remain difficult for end users to refine through direct manipulation, since localized changes

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

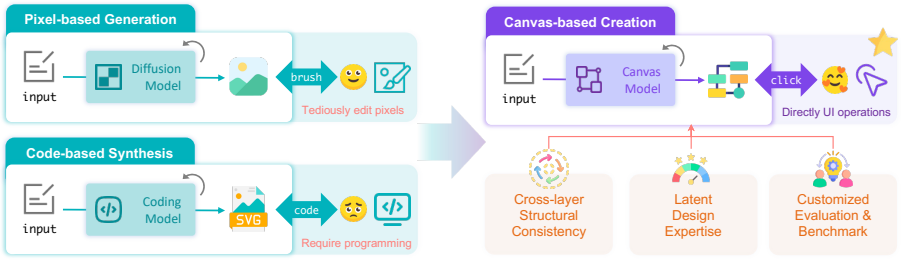


Figure 1: Comparison of diagram generation paradigms. Unlike pixel-based generation (limited control) or code-based synthesis (high barrier), Canvas-based creation unifies AI actionability with human-interpretable UI editing, bridging the representation gap.

often require editing low-level syntax rather than moving, regrouping, or restyling diagram objects. This leaves a representation gap: current formats rarely support autonomous machine generation and intuitive human editing at the same time.

We study editable diagram creation through a canvas-based representation, an intermediate schema composed of object-level entities, relations, styles, and coordinates. This schema is not only an output format but also an action space for generation, verification, refinement, and user manipulation. Because the schema can be rendered into a concrete canvas, visual-spatial defects such as overlap, edge crossing, misalignment, and text overflow become observable after execution. This observation changes the task from one-shot diagram generation into a verifiable design process, where rendered feedback can repair the current artifact and provide reusable experience for future tasks.

Despite this opportunity, realizing canvas-based diagram creation still requires addressing several challenges. First, editable diagram creation requires cross-layer coordination: semantic structure, visual emphasis, and spatial layout are interdependent and cannot be optimized independently. Second, reliable feedback is needed for visual-spatial decisions, since many defects only emerge after the canvas is rendered and inspected. Third, high-quality diagrams depend on latent, domain-dependent design expertise, such as layout conventions, density control, and visual hierarchy, which should be accumulated and retrieved rather than repeatedly rediscovered from scratch.

To address these challenges, we introduce EvoDesign, an agentic framework for editable diagram creation via design expertise evolution. EvoDesign constructs a canvas manifest through a coordinated multi-agent pipeline for semantic structure, visual styling, and spatial layout. It then executes the manifest in a diagnostic verification environment that combines objective programmatic checks with VLM-based critique. The verification output drives a localized refiner to correct the current diagram and provides structured evidence for design expertise evolution. Through this mechanism, EvoDesign distills refinement traces into a three-tier memory of sample strategies, domain guidelines, and general principles, where each abstraction preserves the diagnostic evidence that supports it and can be retrieved by agents in later tasks.

Our contributions are summarized as follows:

- We formulate editable canvas creation as a diagram generation paradigm that bridges pixel-level synthesis and code-level execution through a machine-actionable, user-editable schema.
- We propose verification-guided design expertise evolution, which converts diagnostic refinement traces into a maintained hierarchy of reusable design knowledge.
- We define CanvasBench, a canvas-recoverable benchmark and evaluation protocol with content, visual, and cognitive dimensions for comparing diagram generation methods.

2 Related Work

Automated Diagram Generation. Existing approaches to automated diagram generation mainly rely on two underlying data representations: pixel arrays and symbolic code. Advanced by text-to-image models, pixel-based approaches [27, 16, 37] excel in generating visually diverse renderings. However, these models suffer from stochasticity, frequently producing semantic hallucinations (e.g., illegible text or chaotic relationships) and static, rasterized artifacts. This lack of object-level separability renders post-generation editing impossible without complex inpainting. To ensure structural integrity, research has shifted toward generating executable code. Recent works [3, 24]

focus on synthesizing low-level vector primitives in SVG code to reconstruct intricate geometric details with resolution independence. Alternatively, other works use programmatic languages, such as LaTeX-based TikZ [34, 23] or domain-specific languages like Mermaid [5], which prioritize semantic constraints and structural validity. However, this paradigm introduces a high usability barrier. The resulting representations are brittle for end-users, as localized adjustments require manipulating low-level syntax rather than intuitive visual handles. There lacks a unified representation that is both machine-actionable for generation and human-interpretable for manipulation. EvoDesign addresses this by adopting a canvas-based schema, effectively bridging the gap between the generative precision of symbolic code and the intuitive flexibility of modern design interfaces.

Agentic Media Creation. Multimodal agents now orchestrate complex visual media such as posters, slides, and videos [39, 31, 15]. For paper posters, Paper2Poster [22] and PosterGen [43] use multi-agent systems [35] to condense scientific content into a single visual plane. For multi-page narratives, PPTAgent [44] and SlideTailor [41] automate slide creation by modeling logical flow and structure. For dynamic media, Code2Video [4], Paper2Video [45], and PresentAgent [26] coordinate scripts, visuals, transitions, and voiceovers. While these frameworks treat media as macro-containers for asset arrangement, diagrams are compact, structure-dense kernels [17] that translate abstract semantics into concrete topology within a confined space. Human–AI coding collaboration similarly emphasizes inspectable, revisable shared state [30]. EvoDesign studies agentic creation for this topology-centric medium with an automated expertise evolution perspective.

Agent Memory Evolution. Memory enables agents to move from stateless inference toward continual adaptation [11]. Prior work studies both memory architecture and active evolution [19, 9, 8]: MemoryOS [14] uses a hierarchical framework, A-MEM [38] organizes interconnected knowledge, and MIRIX [33] coordinates memory updates in multi-agent settings. User modeling supports behavior-aware prediction [28], personalized simulation [18], population-aligned persona generation [12], and personalized tutoring and practice [29, 32]. AlphaEvolve [20] evolves code libraries, EvoMem [6] uses dual-evolving memory for planning, ReasoningBank [21] distills reasoning patterns, and ACE [42] treats context as an evolving playbook. High-fidelity diagramming requires implicit expertise over multi-objective visual-spatial trade-offs. EvoDesign therefore distills role-specific design expertise into a hierarchical memory to improve multi-agent synergy and global consistency.

3 Task Formulation

Let $\mathcal{C}$ denote the space of semantic contents providing the informational context, and $\mathcal{R}$ represent optional user requirements specifying preferences for content, style, or layout. We define the diagram generation task as a mapping from c and r to an object-level editable representation D, i.e., $\mathcal{F}_\pi : I = (c, r) \to D$, where π is a design policy, $c \in \mathcal{C}$ and $r \in \mathcal{R} \cup \{\emptyset\}$. Formally, the diagram representation D is defined as $D = (G, S, L)$ with the following components:

- *Semantic Graph* $G = (\mathcal{V}, \mathcal{E})$: A structural abstraction where nodes $\mathcal{V}$ and edges $\mathcal{E}$ capture the logical entities and relational dependencies extracted from the content c.
- *Style Schema* S: A set of visual encodings that assigns attributes (e.g., color, shape, fill) to each element of G based on design principles and user preferences r.
- *Layout Configuration* L: A geometric arrangement on a canvas defining shape dimensions, spatial coordinates, and routing functions to consider perceptual clarity and r.

By executing such a representation within a canvas-based rendering environment (e.g., `excalidraw`), we obtain a visual artifact $\hat{D}$ that is inherently interactive and UI-friendly. The representation is therefore not only an output format, but also an action space for automated generation, diagnostic verification, localized refinement, and direct user manipulation. This allows users to perform post-generation refinements via familiar UI operations while enabling the system to inspect and modify object-level diagram states programmatically. The objective is to synthesize a feasible and effective D^* that balances semantic faithfulness, visual consistency, and spatial resolvability, while satisfying the user requirements r if provided.

4 Agentic EvoDesign

EvoDesign operationalizes editable diagram creation through a design policy $\pi = \{\mathcal{A}, \mathcal{M}\}$, where $\mathcal{A}$ is a coordinated set of generation and refinement agents and $\mathcal{M}$ is an evolving design expertise

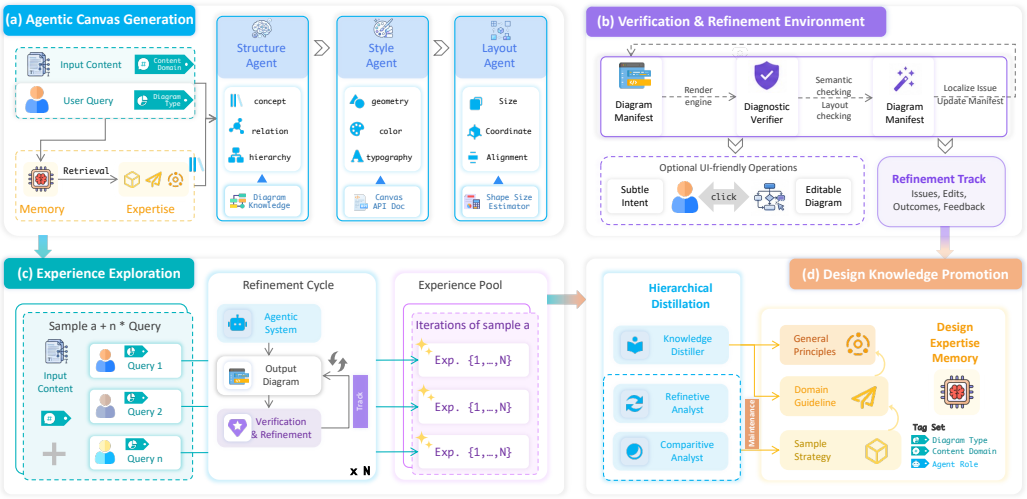


Figure 2: Overview of the EvoDesign framework. *(a) Agentic Canvas Generation*: EvoDesign transforms input content and user queries into a diagram manifest through structure, style, and layout agents, with retrieved expertise and canvas API knowledge grounding the generation process. *(b) Verification and Refinement Environment*: the manifest is rendered into an editable diagram, checked by a diagnostic verifier, and locally repaired through issue localization and manifest updates, while optional UI-friendly operations and refinement tracks record issues, edits, outcomes, and feedback. *(c) Experience Exploration*: multiple query variants for the same content are sampled and repeatedly passed through agentic generation, verification, and refinement to collect comparable experience trajectories. *(d) Design Knowledge Promotion*: refinement tracks are distilled and analyzed into sample strategies, domain guidelines, and general principles, which are maintained with tag sets and stored in the design expertise memory.

memory. Given an input $I = (c, r)$, the system first produces a canvas manifest $D = (G, S, L)$ through multi-agent generation, then renders and verifies the manifest to obtain structured diagnostic feedback. This feedback has two roles: it guides localized repair of the current diagram and provides evidence used to distill reusable design expertise. The overall process therefore links short-term diagram correction with long-term expertise accumulation. Algorithm 1 and Algorithm 2 in Appendix B.1 provide the corresponding pseudocode for the generation loop and the expertise evolution cycle.

4.1 Multi-Agent Canvas Generation

The initial manifest is produced by a coordinated generation backbone with three specialized agents. Each agent follows a spec-to-instance pattern: it first forms a high-level design specification conditioned on the input and retrieved expertise, and then instantiates concrete schema fields that are valid under the canvas renderer. This pattern separates abstract design intent from low-level canvas constraints, reducing hallucinated properties while preserving object-level editability.

Semantic Structure Agent. The structure agent maps the source content into a semantic graph $G = (\mathcal{V}, \mathcal{E})$. It first determines the diagram type, narrative scope, hierarchy, and relational motif needed to express the user intent. It then grounds entities into canonical nodes, removes peripheral noise, nests related concepts into groups, and constructs salient relations. This entity-first, relation-last process provides a stable symbolic topology for downstream styling and layout.

Visual Style Agent. The style agent assigns visual encodings S to nodes, groups, and edges in the semantic graph. It first builds a design specification covering palette, visual hierarchy, shape vocabulary, and emphasis strategy. The specification is then translated into renderer-supported attributes such as color, fill, stroke, and shape type. By grounding abstract style intent into a finite property space, the agent maintains visual consistency and avoids unsupported canvas states.

Spatial Layout Agent. The layout agent computes the geometric configuration L for the styled graph. It uses tool-augmented size estimation to account for text wrapping, group padding, and element footprints before assigning coordinates. The agent then performs recursive placement, resolving

global arrangement for top-level modules and local coordinates within nested groups. This produces an executable layout manifest with positions, dimensions, ordering, and edge routing information.

4.2 Diagnostic Verification and Localized Refinement

Once a manifest is rendered into a canvas artifact $\hat{D}$, EvoDesign evaluates it in a *Diagnostic Verification Environment*. The environment produces two complementary forms of feedback. Objective checks identify programmatic visual-spatial defects, including node overlap, edge crossing, alignment error, text overflow, and canvas utilization. VLM-based critique provides higher-level assessments of semantic completeness, visual hierarchy, and aesthetic coherence that are difficult to express as deterministic rules.

The verification output is represented as a prioritized finding list $\mathcal{F} = \{f_i\}_{i=1}^{n}$. Each finding records the defect category, severity, affected elements, and a natural-language description. Both the objective score S_{obj} and VLM critique score S_{crit} use a 0–10 scale. Objective findings incur a severity-weighted penalty $P = 5n_{\text{critical}} + 3n_{\text{major}} + n_{\text{minor}} + 0.2n_{\text{info}}$, giving $S_{\text{obj}} = 10[1 - \min(P, 50)/50]$; fewer defects yield a higher score. The critique score follows a fixed rubric for semantic and visual quality. In the reported experiments, verification combines these signals and measures refinement improvement as

$$\begin{aligned} \text{score}(\hat{D}) &= 0.6S_{\text{obj}}(\hat{D}) + 0.4S_{\text{crit}}(\hat{D}), \\ \Delta_{\text{score}}^{(t)} &= \text{score}(\hat{D}^{(t+1)}) - \text{score}(\hat{D}^{(t)}). \end{aligned} \tag{1}$$

This score guides local refinement and strategy selection.

The *Localized Refiner Agent* consumes $\mathcal{F}$ and applies targeted edits to the manifest instead of regenerating the whole diagram. Its action space includes single-element operations, such as moving a node or changing a style attribute, and cross-element operations, such as aligning a group or rerouting an edge. After each repair, the diagram is rendered and verified again. This loop improves the current artifact while preserving the semantic structure established by the generation stage.

The finding list and Δ_{score} also support design expertise evolution by linking corrective actions to their observed effects. This generation and verification loop is summarized in Algorithm 1.

4.3 Design Expertise Evolution

Design quality depends on latent visual-spatial heuristics that vary across diagram types, domains, and agent roles. EvoDesign turns such heuristics into reusable expertise through four coupled operations: storing refinement evidence in a hierarchical memory, exploring experience through verification-guided refinement, promoting supported strategies across contexts, and maintaining the resulting memory through usage-aware updates. Algorithm 2 formalizes this evolution cycle.

Hierarchical Expertise Memory. EvoDesign maintains a design expertise memory $\mathcal{M} = \{\mathcal{K}^s, \mathcal{K}^g, \mathcal{K}^p\}$ with three levels of abstraction. The bottom tier $\mathcal{K}^s$ stores *sample strategies*: local, trace-grounded rules that describe how a concrete refinement action resolved a specific diagram failure. The middle tier $\mathcal{K}^g$ stores *domain guidelines*: strategies that receive repeated support across related domains or task contexts. The top tier $\mathcal{K}^p$ stores *general principles*: higher-level design rules that remain useful across diagram types, such as density control, modular separation, or role-consistent visual hierarchy. Each memory item records its tier, responsible agent role, diagram type, domain context, precondition, corrective action, expected effect, evidence identifiers, confidence, and usage statistics. This schema makes each abstraction auditable: a principle can be traced back to the guidelines and sample strategies that supported it, while each local strategy remains tied to its original refinement evidence.

Verification-Guided Experience Exploration. Experience is collected from the same rendered feedback loop used for localized refinement. For a given diagram, vertical exploration follows the refinement trajectory over multiple steps. At each step, the system records an evidence unit containing the pre-repair findings, applied edits, post-repair findings, and Δ_{score}. This step-level record estimates whether a corrective action reduced a concrete visual-spatial failure. To test whether an action remains useful beyond a single prompt, horizontal exploration creates nearby task variants, such as alternative user requirements or contextual constraints for the same content, and runs the generation-refinement loop again. Together, vertical and horizontal evidence turn individual repairs into comparable experience for later expertise evolution.

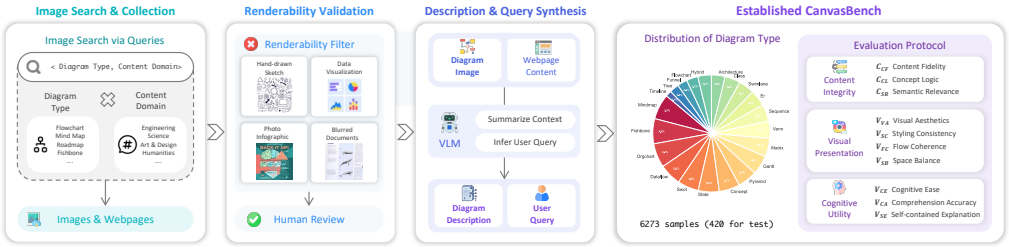


Figure 3: The dataset construction pipeline and the overview of CanvasBench.

Evidence-Guided Expertise Promotion. The evolution process converts refinement experience into increasingly reusable expertise while preserving provenance. At the base tier, EvoDesign extracts candidate strategies from verified refinement traces. Each candidate strategy is linked to the refinement steps, edit operations, and resolved finding categories that support it. Its evidence score is the mean Δ_{score} over its supporting steps, and the candidate enters $\mathcal{K}^s$ only when this mean exceeds $\tau = 0.1$. The mixture weights and τ were fixed on a held-out split before evaluation samples were inspected. Related strategies are then grouped by agent role, diagram context, and action template. Promotion into $\mathcal{K}^g$ requires at least five supporting traces spanning three domains, with links to the source strategies. Promotion into $\mathcal{K}^p$ retains this evidence floor and additionally requires support across at least two diagram types. Promotion therefore expands the scope and confidence of a rule without discarding the evidence needed for later inspection.

Active Knowledge Maintenance. The memory is periodically maintained through scoring, pruning, merging, promotion, and demotion. Confidence combines creation-time evidence from refinement traces with deployment-time evidence from later retrieval outcomes, support coverage, and redundancy checks. Low-utility items are pruned or demoted, semantically redundant items are merged, and high-support items are promoted when later evidence shows broader transferability. During generation, agents retrieve expertise by filtering on agent role, diagram type, and domain, then ranking candidates by semantic relevance and confidence. This retrieval closes the loop: maintained expertise guides future multi-agent generation, while new verified traces update the memory in subsequent evolution rounds.

5 The CanvasBench

Existing diagrammatic benchmarks often contain high-entropy artifacts (e.g., photorealistic textures, complex data visualizations) that are incompatible with vector-based canvas schemas. To bridge this gap, we introduce CanvasBench, a dataset filtered for *diagram editability*, requiring structures that can be represented as editable objects, accompanied by a multi-dimensional evaluation framework spanning content, visual, and cognitive perspectives.

5.1 Data Collection Pipeline

As illustrated in Figure 3, we implement a three-stage pipeline to synthesize diagram-instruction pairs. By intersecting 21 curated diagram types with 30 vertical domains from the MMMU benchmark, we retrieve candidate images and use Qwen3-VL-Max to filter out high-entropy artifacts (e.g., sketches and photo infographics). Two annotators independently review candidate images under written guidelines, with Cohen's $\kappa = 0.74$. URL-level filtering excludes repeated source pages. Image pairs with high embedding similarity are reviewed by a VLM, and low-confidence cases are escalated to human review. No additional near-duplicate images were identified by this review. Finally, a VLM reconstructs dense descriptions and canonical user queries from the validated images and their webpage contexts, producing (image, query, content) triplets. The synthesized descriptions are manually checked for sufficient drawing information, and the description prompts are refined using this feedback. See Appendix D.1 for the search taxonomy and synthesis procedure.

CanvasBench focuses on structured, editable diagrams with reference-derived descriptions and queries.

Table 1: Main results on CanvasBench across content, visual, and cognitive dimensions.

Category	Method	Model	Content			Visual				Cognitive		
			C_{CF}	C_{CL}	C_{SR}	V_{VA}	V_{SC}	V_{FC}	V_{SB}	G_{CE}	G_{CA}	G_{SE}
Diffusion	GPT-4o-Image	–	2.111	2.349	2.444	3.810	3.635	3.651	3.571	4.048	2.270	2.921
Diffusion	Gemini-2.5-Flash-Image	–	1.857	2.127	2.111	3.206	2.889	2.857	3.317	2.889	2.000	2.476
Diffusion	Flux.2 flex	–	1.825	2.000	2.063	3.254	2.952	2.952	3.254	2.937	1.857	2.397
Code-based	LaTeX TikZ	GPT-5.5	3.404	3.421	3.579	2.667	2.579	3.246	2.930	3.772	3.035	2.789
Code-based	LaTeX TikZ	Gemini-3.1-Pro	4.021	3.932	4.018	3.184	3.438	3.512	3.076	3.704	3.352	2.604
Code-based	LaTeX TikZ	DeepSeek-V4-Pro	3.884	3.792	3.946	3.012	3.214	3.281	2.846	3.486	3.186	2.482
Code-based	LaTeX TikZ	DeepSeek-V4-Flash	3.526	3.412	3.561	2.702	2.954	2.986	2.512	3.108	2.714	2.206
Agentic	DiagramAgent	GPT-5.5	2.000	2.476	1.921	2.778	2.825	2.683	2.540	2.349	2.508	2.492
Agentic	DiagramAgent	Gemini-3.1-Pro	2.842	3.102	2.784	3.042	3.168	3.044	2.861	2.934	2.886	2.612
Agentic	DiagramAgent	DeepSeek-V4-Pro	2.716	2.948	2.663	2.884	3.031	2.872	2.702	2.781	2.741	2.486
Agentic	DiagramAgent	DeepSeek-V4-Flash	2.346	2.624	2.281	2.641	2.812	2.563	2.418	2.512	2.438	2.214
Agentic	SciSketch	GPT-5.5	4.048	3.794	3.587	2.984	3.063	2.921	2.841	3.159	3.540	3.222
Agentic	SciSketch	Gemini-3.1-Pro	4.108	3.906	3.784	3.214	3.302	3.182	3.014	3.426	3.612	3.264
Agentic	SciSketch	DeepSeek-V4-Pro	4.006	3.832	3.712	3.072	3.188	3.064	2.902	3.281	3.496	3.146
Agentic	SciSketch	DeepSeek-V4-Flash	3.724	3.514	3.408	2.812	2.986	2.824	2.657	3.014	3.146	2.884
Schema	Prompting	GPT-5.5	3.884	3.566	3.612	2.463	2.891	2.512	2.064	2.806	2.965	2.486
Schema	Prompting	Gemini-3.1-Pro	3.812	3.741	3.829	2.752	3.116	2.866	2.351	3.168	3.034	2.604
Schema	Prompting	DeepSeek-V4-Pro	3.738	3.664	3.758	2.557	2.998	2.674	2.136	2.914	2.898	2.376
Schema	Prompting	DeepSeek-V4-Flash	3.438	3.274	3.371	2.276	2.742	2.412	1.886	2.586	2.482	2.008
Canvas-native	**EvoDesign**	GPT-5.5	4.118	4.052	4.126	3.612	3.786	3.704	3.536	3.912	3.512	3.086
Canvas-native	**EvoDesign**	Gemini-3.1-Pro	4.062	4.018	4.097	3.526	3.714	3.681	3.472	3.861	3.486	3.054
Canvas-native	**EvoDesign**	DeepSeek-V4-Pro	3.936	3.887	4.012	3.317	3.562	3.421	3.213	3.674	3.342	2.891
Canvas-native	**EvoDesign**	DeepSeek-V4-Flash	3.684	3.602	3.758	3.038	3.326	3.082	2.842	3.386	3.012	2.548

5.2 Dataset Statistics

Finally, CanvasBench contains 6,273 validated samples across 21 diagram types, consisting of 5,853 samples for training and design expertise evolution and a balanced held-out test set of 420 samples (20 per diagram type). Each sample ensures object-level canvas editability on diverse, real-world visual topologies. As illustrated in Figure 3, the training/evolution split preserves broad coverage over diagram categories, while the test split controls for type frequency to support comparable evaluation. The samples preserve heterogeneous layout styles from human-authored diagrams. Further analysis and examples appear in Appendix D.3 and Appendix D.4.

5.3 Evaluation Protocol

Rigorous diagram assessment extends beyond semantic correctness and visual presentation to ensure *cognitive interpretability*. To capture this, we propose a multi-dimensional evaluation framework structured along three primary axes. Detailed definitions are provided in Appendix D.5. VLM-based judges assign scores on a 5-point Likert scale.

- *Content Integrity Dimension ($\mathcal{C}$)* evaluates semantic alignment between the source text and the generated schema via *Content Fidelity* (C_{CF}), Concept Logic (C_{CL}), and Semantic Relevance (C_{SR}).
- *Visual Presentation Dimension ($\mathcal{V}$)* assesses the structural and aesthetic validity of the canvas style and layout through Visual Aesthetics (V_{VA}), Styling Consistency (V_{SC}), Flow Coherence (V_{FC}), and Space Balance (V_{SB}).
- *Cognitive Utility Dimension ($\mathcal{G}$)* quantifies the cognitive efficiency and interpretability for information translation using Cognitive Ease (G_{CE}), Comprehension Accuracy (G_{CA}), and Self-contained Explanation (G_{SE}).

Cognitive Ease estimates the effort required to read the visual structure, while Comprehension Accuracy uses VLM-based question answering to check whether factual and structural information can be recovered solely from the diagram. Self-contained Explanation evaluates whether the diagram conveys its narrative without access to the source text.

6 Experiments

In this section, we evaluate EvoDesign against representative baselines on CanvasBench.

6.1 Experimental Settings

Evaluation Metrics. We adopt the *CanvasBench* evaluation protocol defined in Section 5.3, where VLM-based judges assign scores on a 5-point Likert scale across content, visual, and cognitive dimensions. For the cross-model rows, we report the mean score of each metric over the balanced evaluation subset.

Comparative Baselines. We compare against pixel-based generators, LaTeX TikZ code synthesis, one-shot canvas-schema generation, and agentic diagram systems. For the non-pixel baselines, we evaluate the same method under GPT-5.5, Gemini-3.1-Pro, DeepSeek-V4-Pro, and DeepSeek-V4-Flash whenever the method can use a backbone VLM. All methods are evaluated on the same balanced CanvasBench subset, with task prompts normalized to expose the same source content and diagram intent. EvoDesign uses the `excalidraw` canvas schema as the executable representation, with the agent workflow orchestrated by LANGGRAPH. Unless otherwise stated, generation uses temperature 0.7, top-k retrieval from the design expertise memory, and one localized refinement pass after diagnostic feedback. Additional implementation details, data splits, and baseline settings are provided in Appendix C.

6.2 Main Results

EvoDesign exhibits a balanced performance profile across content, visual, and cognitive axes. As shown in Table 1, code-based and agentic baselines can be strong on selected content metrics, especially LaTeX TikZ and SciSketch under stronger backbones. However, they remain less consistent on canvas-native visual organization and cognitive usability. By contrast, EvoDesign achieves the strongest overall balance across the four backbone models. Its most pronounced gains appear on styling fitness, space balance, cognitive ease, and self-contained explanation, matching the goal of editable diagram creation rather than static image fidelity alone.

The cross-model rows also show that the improvement is not tied to a single VLM. Under GPT-5.5, EvoDesign improves over one-shot prompting by $+1.472$ on Space Balance and $+1.106$ on Cognitive Ease. The same pattern holds for Gemini-3.1-Pro, DeepSeek-V4-Pro, and DeepSeek-V4-Flash, where EvoDesign remains consistently ahead of prompting and competitive external baselines on the visual and cognitive dimensions.

6.3 Ablation Study

We conduct an ablation study in Table 2 to isolate the multi-agent decomposition, design expertise memory, and localized refiner across backbone models. The w/o Multiagent setting replaces role-specialized structure, style, and layout planning with a less decomposed generation process; w/o Memory disables retrieved design expertise; and w/o Refiner removes the localized repair step after diagnostic feedback. Across all backbones, the full system consistently outperforms these variants. The largest drops appear when removing multi-agent decomposition, especially on Space Balance, while memory and refinement provide complementary gains on reusable design priors and local visual repair.

6.4 Knowledge Evolution Dynamics

We track validation quality and memory growth over five evolution rounds to assess the cumulative benefit of design expertise evolution. The evolution curve shows that validation quality increases steadily as the system accumulates verified refinement traces, while the memory expands from local sample strategies into higher-level guidelines and principles. This trend supports the role of the evolution mechanism as a reusable design-knowledge accumulator for later diagram generation.

6.5 Qualitative and Human-Facing Analysis

Figure 5 shows a memory-return edge crossing Critique and Revise in a research-assistant workflow. Rerouting this edge around the reflection branch clears the intersections while preserving the nodes, labels, and directed connections. The complete workflow and cross-paradigm generation comparison appear in Appendix E.2 and Appendix E.1. The web application exposes the same manifest as object-level canvas elements. Users can create or reopen projects, access design-memory controls,

Table 2: Ablation results across backbone models.

Model	Variant	Content			Visual				Cognitive		
		C_{CF}	C_{CL}	C_{SR}	V_{VA}	V_{SC}	V_{FC}	V_{SB}	G_{CE}	G_{CA}	G_{SE}
GPT-5.5	**EvoDesign**	4.118	4.052	4.126	3.612	3.786	3.704	3.536	3.912	3.512	3.086
GPT-5.5	*w/o Multiagent*	3.946	3.901	3.982	3.184	3.426	3.298	2.824	3.512	3.246	2.812
GPT-5.5	*w/o Memory*	4.036	3.972	4.041	3.428	3.566	3.482	3.178	3.684	3.402	2.938
GPT-5.5	*w/o Refiner*	4.074	4.006	4.083	3.361	3.498	3.548	3.096	3.642	3.431	2.951
Gemini-3.1-Pro	**EvoDesign**	4.062	4.018	4.097	3.526	3.714	3.681	3.472	3.861	3.486	3.054
Gemini-3.1-Pro	*w/o Multiagent*	3.912	3.887	3.962	3.126	3.338	3.245	2.752	3.476	3.214	2.781
Gemini-3.1-Pro	*w/o Memory*	3.976	3.941	4.006	3.348	3.472	3.441	3.136	3.621	3.367	2.872
Gemini-3.1-Pro	*w/o Refiner*	4.018	3.972	4.041	3.286	3.421	3.512	3.048	3.584	3.402	2.884
DeepSeek-V4-Pro	**EvoDesign**	3.936	3.887	4.012	3.317	3.562	3.421	3.213	3.674	3.342	2.891
DeepSeek-V4-Pro	*w/o Multiagent*	3.811	3.786	3.892	2.952	3.171	3.018	2.541	3.247	3.056	2.612
DeepSeek-V4-Pro	*w/o Memory*	3.872	3.831	3.936	3.156	3.294	3.217	2.941	3.426	3.204	2.718
DeepSeek-V4-Pro	*w/o Refiner*	3.918	3.846	3.954	3.091	3.238	3.263	2.864	3.392	3.241	2.731
DeepSeek-V4-Flash	**EvoDesign**	3.684	3.602	3.758	3.038	3.326	3.082	2.842	3.386	3.012	2.548
DeepSeek-V4-Flash	*w/o Multiagent*	3.556	3.498	3.627	2.674	2.952	2.684	2.236	3.027	2.718	2.238
DeepSeek-V4-Flash	*w/o Memory*	3.621	3.546	3.689	2.862	3.084	2.904	2.612	3.164	2.884	2.371
DeepSeek-V4-Flash	*w/o Refiner*	3.672	3.562	3.701	2.791	3.036	2.942	2.514	3.132	2.912	2.386

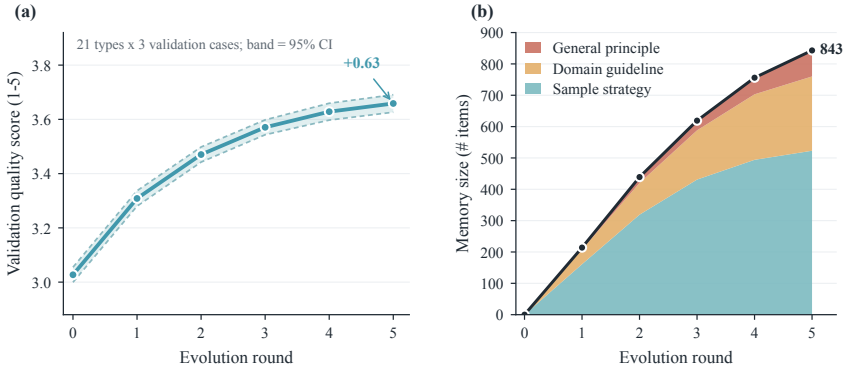


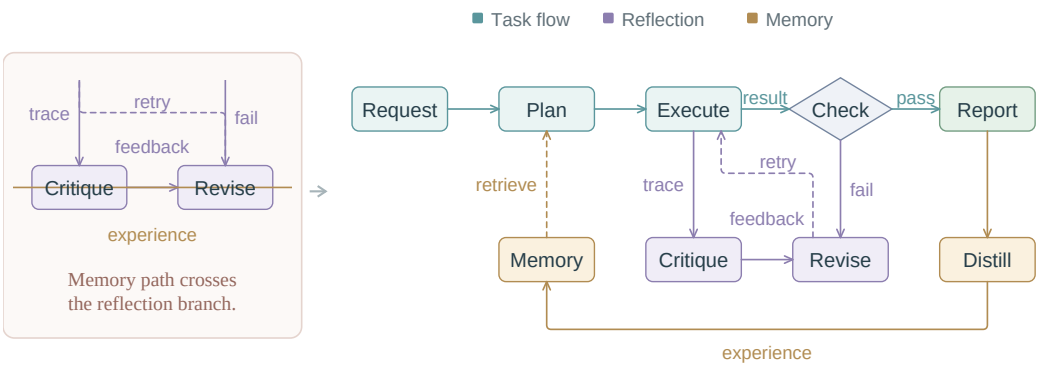
Figure 4: Evolution dynamics of the design expertise memory. Left: validation quality improves as refinement traces accumulate across evolution rounds. Right: the memory grows from sample-level strategies into reusable domain guidelines and general principles.

and adjust element-level properties through direct manipulation. A selection-scoped design assistant accepts natural-language requests for local edits, allowing users to combine automated refinement with manual changes on the same canvas (Appendix F).

6.6 Human Study

To complement automatic evaluation, we conducted a user study with 36 participants who had prior experience in diagram authoring. Participants used the web application for their own diagramming tasks at least ten times within a day before completing eight questionnaire items on a five-point Likert scale (1 = strongly disagree, 5 = strongly agree). The items cover personalized memory and editable refinement; the protocol is provided in Appendix F.

As shown in Figure 6, participants gave the highest rating to requiring fewer edits (4.4 ± 0.6). They also reported that the system learned their preferences (4.3 ± 0.8), preserved source meaning during refinement (4.3 ± 0.7), and provided a preferred editing workflow (4.2 ± 0.8). These ratings indicate positive perceived personalization and editing quality.



(a) Initial detail **(b) Refined workflow**

Figure 5: Localized refinement of a research-assistant workflow. Rerouting the memory return clears the reflection branch while preserving all nine nodes and eleven directed connections.

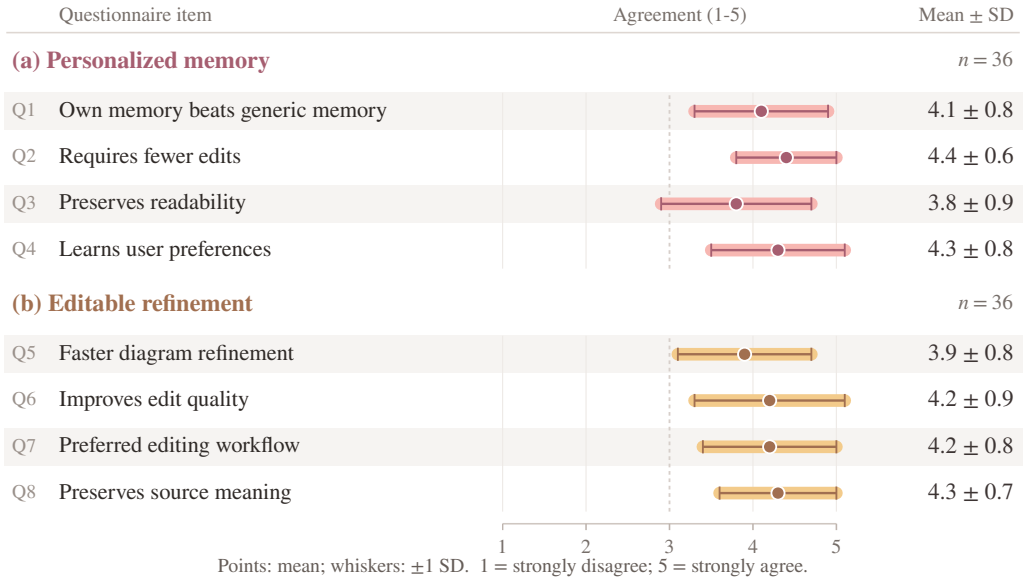


Figure 6: Self-reported user experience with EvoDesign ($n = 36$). Points show mean agreement and horizontal whiskers show one standard deviation. Panels summarize (a) personalized memory and (b) editable refinement.

7 Conclusion

In this paper, we introduced EvoDesign, an agentic framework designed to bridge the representation gap in automated diagramming. By generating diagrams through a canvas-based schema rather than static pixels or brittle code, EvoDesign enables a fluid transition between autonomous AI generation and human-centric UI intervention. The framework addresses multi-layer diagramming constraints through a coordinated multi-agent pipeline and a design knowledge evolution mechanism that can organize verified refinement traces into reusable expertise. To support rigorous research in this paradigm, we define CanvasBench, a dataset construction process and evaluation protocol focused on object-level editability and cognitive utility. Experiments across pixel-based, code-based, agentic, one-shot schema, and canvas-native methods show that EvoDesign provides a balanced advantage on visual organization and cognitive usability while remaining competitive on content fidelity. In a user study with 36 participants, users reported positive experiences with personalized memory and editable refinement.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant No. 92370204); in part by the National Key R&D Program of China (Grant No. 2023YFF0725001); in part by the Guangdong Provincial Key Laboratory of Frontier Basic Science for All-domain Intelligence; in part by the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2023B1515120057); and in part by the Key-Area Special Project of Guangdong Provincial Ordinary Universities (2024ZDZX1007).

References

[1] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[2] Fengxiang Bie, Yibo Yang, Zhongzhu Zhou, Adam Ghanem, Minjia Zhang, Zhewei Yao, Xiaoxia Wu, Connor Holmes, Pareesa Golnari, David A Clifton, et al. Renaissance: A survey into ai text-to-image generation in the era of large model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[3] Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. Deepsvg: A hierarchical generative network for vector graphics animation. *Advances in Neural Information Processing Systems*, 33:16351–16361, 2020.

[4] Yanzhe Chen, Kevin Qinghong Lin, and Mike Zheng Shou. Code2video: A code-centric paradigm for educational video generation. *arXiv preprint arXiv:2510.01174*, 2025.

[5] Pritam Deka and Barry Devereux. Flowchart2mermaid: A vision-language model powered system for converting flowcharts into editable diagram code. *arXiv preprint arXiv:2512.02170*, 2025.

[6] Wenzhe Fan, Ning Yan, and Masood Mortazavi. Evomem: Improving multi-agent planning with dual-evolving memory. *arXiv preprint arXiv:2511.01912*, 2025.

[7] 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.

[8] Linxiao Gong, Yijie Xu, Tianfu Wang, Yin Wu, Yili Wang, Xingbo Yao, Huizai Yao, Xilin Xia, Haowen Yang, and Hui Xiong. Towards communication-efficient social intelligence in language agents. *arXiv preprint arXiv:2609.35749*, 2026. doi: 10.48550/arXiv.2609.35749. URL `https://arxiv.org/abs/2609.35749`.

[9] Zhezheng Hao, Tianfu Wang, Huanshuo Dong, Ziyan Liu, Hong Wang, Xiankun Lin, Qiang Lin, Can Wang, Hande Dong, and Jiawei Chen. Evolve as a team: Collaborative self-evolution for LLM-based multi-agent systems. *arXiv preprint arXiv:2605.29790*, 2026. URL `https://arxiv.org/abs/2605.29790`.

[10] Kairui Hu, Penghao Wu, Fanyi Pu, Wang Xiao, Yuanhan Zhang, Xiang Yue, Bo Li, and Ziwei Liu. Video-mmmu: Evaluating knowledge acquisition from multi-discipline professional videos. *arXiv preprint arXiv:2501.13826*, 2025.

[11] Yuyang Hu, Shichun Liu, Yanwei Yue, Guibin Zhang, Boyang Liu, Fangyi Zhu, Jiahang Lin, Honglin Guo, Shihan Dou, Zhiheng Xi, et al. Memory in the age of ai agents. *arXiv preprint arXiv:2512.13564*, 2025.

[12] Zhengyu Hu, Jianxun Lian, Zheyuan Xiao, Max Xiong, Tianfu Wang, Teng Xiao, Yuxuan Lei, Fengqing Jiang, Kaize Ding, Ziang Xiao, Nicholas Jing Yuan, Xing Xie, and Radha Poovendran. Population-aligned persona generation for LLM-based social simulation. *arXiv preprint arXiv:2509.10127*, 2025. doi: 10.48550/arXiv.2509.10127. URL `https://arxiv.org/abs/2509.10127`.

[13] Thomas Kamps. *Diagram design: a constructive theory*. Springer Science & Business Media, 2012.

[14] Jiazheng Kang, Mingming Ji, Zhe Zhao, and Ting Bai. Memory os of ai agent. *arXiv preprint arXiv:2506.06326*, 2025.

[15] Kimi Team. Kimi K3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026. doi: 10.48550/arXiv.2607.24653. URL `https://arxiv.org/abs/2607.24653`.

[16] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

[17] Jill H Larkin and Herbert A Simon. Why a diagram is (sometimes) worth ten thousand words. *Cognitive science*, 11(1):65–100, 1987.

[18] Yuxuan Lei, Tianfu Wang, Jianxun Lian, Zhengyu Hu, Defu Lian, and Xing Xie. Human-LLM: Towards personalized understanding and simulation of human nature. *arXiv preprint arXiv:2601.15793*, 2026. doi: 10.48550/arXiv.2601.15793. URL `https://arxiv.org/abs/2601.15793`.

[19] Haochen Luo, Yi Huang, Sichun Luo, Fengyuan Liu, Lei Li, Zefa Hu, Junlan Feng, and Qi Liu. Harness-aware self-evolving: Co-evolving model weights, harness, and task solutions. *arXiv preprint arXiv:2607.03935*, 2026. doi: 10.48550/arXiv.2607.03935. URL `https://arxiv.org/abs/2607.03935`.

[20] Alexander Novikov et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

[21] Siru Ouyang, Jun Yan, I Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T Le, Samira Daruki, Xiangru Tang, et al. Reasoningbank: Scaling agent self-evolving with reasoning memory. *arXiv preprint arXiv:2509.25140*, 2025.

[22] Wei Pang, Kevin Qinghong Lin, Xiangru Jian, Xi He, and Philip Torr. Paper2poster: Towards multimodal poster automation from scientific papers. *Advances in Neural Information Processing Systems*, 2025.

[23] Charly Reux, Mathieu Acher, Djamel Eddine Khelladi, Clément Quinton, and Olivier Barais. Llm code customization with visual results: A benchmark on tikz. In *Proceedings of the 29th International Conference on Evaluation and Assessment in Software Engineering*, pages 1086–1096, 2025.

[24] Juan A Rodriguez, Abhay Puri, Shubham Agarwal, Issam H Laradji, Pau Rodriguez, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. Starvector: Generating scalable vector graphics code from images and text. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16175–16186, 2025.

[25] Juan A Rodriguez, Abhay Puri, Shubham Agarwal, Issam H Laradji, Pau Rodriguez, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. Starvector: Generating scalable vector graphics code from images and text. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16175–16186, 2025.

[26] Jingwei Shi, Zeyu Zhang, Biao Wu, Yanjie Liang, Meng Fang, Ling Chen, and Yang Zhao. Presentagent: Multimodal agent for presentation video generation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 760–773, 2025.

[27] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[28] Tianfu Wang, Liwei Deng, Chao Wang, Jianxun Lian, Yue Yan, Nicholas Jing Yuan, Qi Zhang, and Hui Xiong. COMET: NFT price prediction with wallet profiling. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5893–5904. Association for Computing Machinery, 2024. doi: 10.1145/3637528.3671621. URL `https://doi.org/10.1145/3637528.3671621`.

[29] Tianfu Wang, Yi Zhan, Jianxun Lian, Zhengyu Hu, Nicholas Jing Yuan, Qi Zhang, Xing Xie, and Hui Xiong. Llm-powered multi-agent framework for goal-oriented learning in intelligent tutoring system. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 510–519, 2025.

[30] Tianfu Wang, Zhezheng Hao, Yin Wu, Wei Wu, Haowen Yang, Linxiao Gong, Qiang Lin, Hande Dong, Nicholas Jing Yuan, and Hui Xiong. Scaling human-AI coding collaboration requires a governable consensus layer. *arXiv preprint arXiv:2604.17883*, 2026. URL `https://arxiv.org/abs/2604.17883`.

[31] Tianfu Wang, Zhezheng Hao, Xilin Xia, Lixin Liu, Mengkang Hu, Hongzhang Liu, Xi Chen, Ziyan Liu, Xiankun Lin, Weijia Zhang, Nicholas Jing Yuan, and Hui Xiong. Agentic artifact creation: Systems, evaluation, principles, and opportunities. *arXiv preprint arXiv:2608.28122*, 2026. URL `https://arxiv.org/abs/2608.28122`.

[32] Tianfu Wang, Max Xiong, Jianxun Lian, Hongyuan Zhu, Zhengyu Hu, Yuxuan Lei, Linxiao Gong, Dapeng Hu, Xiaofang Li, Peiting Tsai, Nicholas Jing Yuan, and Qi Zhang. Social-Coach: Personalized social skill learning with agentic tutoring and practice. *arXiv preprint arXiv:2606.04155*, 2026. URL `https://arxiv.org/abs/2606.04155`.

[33] Yu Wang and Xi Chen. Mirix: Multi-agent memory system for llm-based agents. *arXiv preprint arXiv:2507.07957*, 2025.

[34] Jingxuan Wei, Cheng Tan, Qi Chen, Gaowei Wu, Siyuan Li, Zhangyang Gao, Linzhuang Sun, Bihui Yu, and Ruifeng Guo. From words to structured visuals: A benchmark and framework for text-to-diagram generation and editing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13315–13325, 2025.

[35] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*.

[36] Tsung-Han Wu, Long Lian, Joseph E Gonzalez, Boyi Li, and Trevor Darrell. Self-correcting llm-controlled diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6327–6336, 2024.

[37] Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024.

[38] Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *Advances in Neural Information Processing Systems*, 2025.

[39] Huanjin Yao, Ruifei Zhang, Jiaxing Huang, Jingyi Zhang, Yibo Wang, Bo Fang, Ruolin Zhu, Yongcheng Jing, Shunyu Liu, Guanbin Li, et al. A survey on agentic multimodal large language models. *arXiv preprint arXiv:2510.10991*, 2025.

[40] Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567, 2024.

[41] Wenzheng Zeng, Mingyu Ouyang, Langyuan Cui, and Hwee Tou Ng. Slidetailor: Personalized presentation slide generation for scientific papers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[42] Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*, 2025.

[43] Zhilin Zhang, Xiang Zhang, Jiaqi Wei, Yiwei Xu, and Chenyu You. Postergen: Aesthetic-aware paper-to-poster generation via multi-agent llms. *arXiv preprint arXiv:2508.17188*, 2025.

[44] Hao Zheng, Xinyan Guan, Hao Kong, Wenkai Zhang, Jia Zheng, Weixiang Zhou, Hongyu Lin, Yaojie Lu, Xianpei Han, and Le Sun. Pptagent: Generating and evaluating presentations beyond text-to-slides. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 14413–14429, 2025.

[45] Zeyu Zhu, Kevin Qinghong Lin, and Mike Zheng Shou. Paper2video: Automatic video generation from scientific papers. *arXiv preprint arXiv:2510.05096*, 2025.

A Further Discussion

A.1 Limitations

Canvas Schema Expressivity. The canvas schema supports general-purpose diagrams such as flowcharts, entity-relationship models, concept maps, and system architectures. Support is more limited for dense scientific visualizations, complex mathematical notation, and artistic illustrations.

Cold-Start Behavior and Cross-Model Generalization. Memory quality depends on the number and diversity of accumulated refinement traces. Cold-start stability and transfer across backbone model families require further study.

Benchmark Input Construction and Selection. CanvasBench's reference-derived inputs can specify structure more explicitly than independently authored requests, and its editability filter favors structured diagrams. Less structured artifacts and naturally authored requests are therefore underrepresented.

Computational Cost. Each diagram generation incurs multiple LLM calls for semantic structure, visual style, spatial layout, and refinement. The knowledge evolution phase adds additional distillation calls per sample. These calls increase interactive latency; smaller-model routing and output caching could reduce the cost.

A.2 Broader Impacts

EvoDesign lowers the barrier to producing structured visual artifacts, which can help non-experts communicate complex information across education, business, science, and engineering workflows. Its canvas-based representation also supports human-AI co-creation by keeping generated artifacts object-level editable rather than static raster images. Incomplete or incorrectly parsed source content can produce misleading diagrams, making human review important. CanvasBench also uses web-derived visual materials during dataset construction, so release and redistribution should respect source attribution, licensing, and privacy constraints.

A.3 LLM Usage

We use LLMs and VLMs as core components of the proposed system, including diagram manifest generation, diagnostic critique, localized refinement, evaluation protocol design, and design expertise distillation. We also used general-purpose language models during paper development for editing assistance, grammar checking, and concise wording suggestions.

A.4 Code and Artifacts

Code and prompts are available at `https://github.com/GeminiLight/EvoDesign`.

Algorithm 1 Diagram Generation with Closed-Loop Refinement

Require: Content c, optional requirements r, knowledge memory $\mathcal{M}$, max refinement iterations $T_{\max}$ (set to 1 in the main experiments)
Ensure: Rendered diagram artifact $\hat{\mathcal{D}}'$, refinement trace $\mathcal{T}$

Retrieve context-aware heuristics $\mathcal{K} \subset \mathcal{M}$ based on task metadata
$G \leftarrow A_{\text{str}}(\mathcal{K}_s, c, r)$ {Semantic Structure Agent}
$S \leftarrow A_{\text{sty}}(\mathcal{K}_{\text{sty}}, c, r, G)$ {Visual Style Agent}
$L \leftarrow A_{\text{lay}}(\mathcal{K}_{\text{lay}}, c, r, G, S)$ {Spatial Layout Agent}
$\mathcal{D} \leftarrow (G, S, L)$ {Compose canvas manifest}
$\hat{\mathcal{D}} \leftarrow \textsc{Render}(\mathcal{D})$ {Render to canvas artifact}
$\mathcal{T} \leftarrow \emptyset$ {Refinement evidence trace}
for $t = 1$ **to** $T_{\max}$ **do**
  $\mathcal{F} \leftarrow \textsc{Diagnose}(\hat{\mathcal{D}}, c)$ {Objective checks + VLM critique}
  if $\mathcal{F} = \emptyset$ **then**
    break {No defects found}
  end if
  $\hat{\mathcal{D}}' \leftarrow A_{\text{ref}}(\hat{\mathcal{D}}, \mathcal{F})$ {Localized Refiner Agent}
  $\mathcal{F}' \leftarrow \textsc{Diagnose}(\hat{\mathcal{D}}', c)$
  $\Delta^{(t)}_{\text{score}} \leftarrow \textsc{Score}(\hat{\mathcal{D}}') - \textsc{Score}(\hat{\mathcal{D}})$
  Record evidence $e^{(t)} = (\mathcal{F}, a^{(t)}, \mathcal{F}', \Delta^{(t)}_{\text{score}})$
  $\mathcal{T} \leftarrow \mathcal{T} \cup \{e^{(t)}\}$
  $\hat{\mathcal{D}} \leftarrow \hat{\mathcal{D}}'$
end for
return $\hat{\mathcal{D}}, \mathcal{T}$

B System Implementation Details

B.1 Algorithmic Overview

Algorithm 1 summarizes the end-to-end diagram generation pipeline, including multi-agent coordination and the closed-loop refinement mechanism. Algorithm 2 details the progressive expertise evolution cycle that distills verified traces into a maintained hierarchical design memory.

B.2 Specification Dimension Summarization of Agents

The Semantic Structure Agent specifies the diagram type, visual rationale, primary flow, and grouping before instantiating the semantic graph.

- *Selected Diagram Type* represents the precise visualization archetype chosen from the knowledge base, such as a Flowchart, Mindmap, and Block Diagram, to align the structural skeleton with the underlying patterns of the text.
- *Visual Rationale* establishes a concise mental blueprint that governs the narrative scope and serves as a semantic filter to exclude high-entropy noise.
- *Primary Flow Strategy* dictates the global directional logic such as linear propagation to ensure the synthesized topology remains legible and logically sound.
- *Grouping Logic* defines the boundaries of organizational containers to aggregate related entities and maintain relational proximity within the semantic structure.

The Visual Style Agent specifies visual tone, hierarchy, color, shapes, and connector styles.

- *Visual Tone Principle* determines the overall artistic character of the diagram by utilizing specific font and line styles to create a professional atmosphere. *Hierarchy Principle* formulates a strategy for visual prioritization by managing the sequential rank index of elements to ensure a clear and organized depth of information.
- *Color Principle* maps distinct palettes to elements based on their categorical meaning to enhance thematic coherence across the entire canvas.

Algorithm 2 Progressive Design Expertise Evolution

Require: Training samples $\{I_i\}_{i=1}^{N}$, evolution rounds R, strategy threshold $\tau = 0.1$, knowledge memory $\mathcal{M} = \{\mathcal{K}^s, \mathcal{K}^g, \mathcal{K}^p\}$
Ensure: Updated memory $\mathcal{M}$

for round $r = 1$ **to** R **do**
  Sample batch $\mathcal{B} \subset \{I_i\}_{i=1}^{N}$
  for each task instance $I \in \mathcal{B}$ **do**
    Retrieve $\mathcal{K} \subset \mathcal{M}$ based on task metadata of I
    $\mathcal{T}_{\text{vert}} \leftarrow$ Run Algorithm 1 on I and collect vertical refinement evidence
    $\{I'_j\}_{j=1}^{m} \leftarrow$ Generate requirement or context variants for I
    $\mathcal{T}_{\text{horiz}} \leftarrow \emptyset$
    for each variant I'_j **do**
      $\mathcal{T}_{\text{horiz}} \leftarrow \mathcal{T}_{\text{horiz}} \cup$ Run Algorithm 1 on I'_j
    end for
    $\mathcal{T}_I \leftarrow \mathcal{T}_{\text{vert}} \cup \mathcal{T}_{\text{horiz}}$
    $\mathcal{C}_I \leftarrow$ Extract candidate strategies from $\mathcal{T}_I$ with evidence links
    for each candidate strategy c with supporting evidence $E_c \subset \mathcal{T}_I$ **do**
      $q_c \leftarrow |E_c|^{-1} \sum_{e \in E_c} \Delta_{\text{score}}(e)$ {Mean linked-step improvement}
      if $q_c > \tau$ **then**
        $k^s \leftarrow$ Distill a sample strategy with precondition, action, effect, tags, confidence q_c, and provenance E_c
        $\mathcal{K}^s \leftarrow \mathcal{K}^s \cup \{k^s\}$
      else if $q_c < -\tau$ **then**
        Record the trace as low-utility evidence for pruning or demotion
      end if
    end for
  end for
  Group related strategies by agent role, diagram type, domain, and action template
  Promote strategy groups supported by ≥ 5 traces across ≥ 3 domains into $\mathcal{K}^g$
  Promote guideline groups retaining this support and spanning ≥ 2 diagram types into $\mathcal{K}^p$
  $\mathcal{M} \leftarrow \textsc{Score}(\mathcal{M})$
  $\mathcal{M} \leftarrow \textsc{Prune}(\mathcal{M})$
  $\mathcal{M} \leftarrow \textsc{Merge}(\mathcal{M})$
  $\mathcal{M} \leftarrow \textsc{Promote}(\mathcal{M})$
  $\mathcal{M} \leftarrow \textsc{Demote}(\mathcal{M})$
end for
return $\mathcal{M}$

- *Shape Principle* selects geometric primitives that visually communicate the functional role of each element to ensure intuitive recognition.
- *Connection Principle* prescribes the visual defaults for relational edges including arrowhead styles and stroke colors to minimize visual clutter.

The Spatial Layout Agent specifies flow direction, spacing, alignment, layering, and nested organization.

- *Flow Strategy Principle* governs the global arrangement of components based on relational dependencies to determine the optimal directional orientation.
- *Spatial Breathing Principle* regulates the margin and padding logic by employing a progressive tightening strategy to ensure sufficient whitespace around complex groups.
- *Alignment Balance Principle* organizes parallel or sibling items into balanced grids to account for the physical footprint of text-heavy shapes.
- *Depth Layering Principle* manages the vertical stacking order to ensure that nested children are rendered on top of parent backgrounds without occlusion.
- *Structural Rhythm Principle* optimizes space utilization by alternating between horizontal flows for outer modules and vertical lists for inner details.

B.3 Predefined Diagram Types in EvoDesign

We manually curated twenty-one diagram types specifically tailored for canvas editability. Table 3 summarizes the key characteristics and optimal applications for these selected types.

Table 3: Comprehensive Summary of Diagram Types.

Chart Type	Best For	Key Characteristics
Architecture Diagram	System design, software structure	Modules, components, and interfaces
Class Diagram	OOP modeling, database schemas	Classes, attributes, methods, inheritance
Concept Map	Knowledge organization, learning	Nodes connected by labeled relationships
Data Flow Diagram	System information flow analysis	Processes, data stores, external entities
Entity Relationship	Database design, modeling data	Entities, attributes, relationships (1:N, M:N)
Fishbone Diagram	Root cause analysis	Spine (problem) with ribs (causes)
Flowchart	Process steps, decision logic	Sequential shapes with directional arrows
Funnel Chart	Sales pipelines, conversion rates	Tapering stages showing reduction
Gantt Chart	Project scheduling, timelines	Horizontal bars representing tasks over time
Hybrid Diagram	Complex multi-view systems	Combination of elements from various diagrams
Matrix Chart	Multi-variable comparisons	Grid structure intersecting data points
Mind Map	Brainstorming, ideation	Central topic with radiating subbranches
Org Chart	Hierarchy, reporting lines	Tree structure of roles and ranks
Pyramid Chart	Hierarchical levels, priority	Stacked triangular layers
Sequence Diagram	Interaction ordering, messaging	Vertical lifelines with time-based messages
State Diagram	System states, transitions	Nodes triggered by events
Swimlane Diagram	Cross-functional workflows	Parallel lanes distinguishing responsibilities
SWOT Analysis	Strategic planning	2x2 matrix (Strengths, Weaknesses, etc.)
Timeline	Chronological history, events	Linear axis marked with dates/milestones
Tree Diagram	Decomposition, probability	Root node branching into hierarchical children
Venn Diagram	Set relationships, logic overlaps	Overlapping circles sharing attributes

B.4 Canvas Schema Space in Excalidraw Library

The Excalidraw schema represents diagrams through styling and layout properties. Table 4 lists shapes, colors, strokes, and typography; Table 5 lists coordinates, dimensions, and layer order. Agents instantiate these properties to produce editable canvas objects.

B.5 Auxiliary Layout Support in EvoDesign

B.5.1 Geometric Size Estimator for Layout Agent

The Geometric Size Estimator estimates minimum node bounds from text wrapping, font properties, padding, and icon space. Dimensions are rounded up to integer values before recursive placement.

Table 4: Styling-related properties and categorical parameter spaces of `excalidraw` shape elements

Category	Property	Candidate Parameters Values
Color	`TLColor`	black, grey, light-violet, violet, blue, light-blue, yellow, orange, green, light-green, light-red, red, white
Geometry	`TLGeo`	rectangle, ellipse, triangle, diamond, pentagon, hexagon, octagon, star, cloud, arrowLeft, arrowRight, arrowUp, arrowDown
Size	`TLSize`	s, m, l, xl
Fill Style	`TLFill`	none, semi, solid, pattern
Dash Style	`TLDash`	draw, solid, dashed, dotted
Typography	`TLFont`	draw, sans, serif, mono

Table 5: Layout-related properties and discretized parameter spaces of `excalidraw` shape elements

Category	Property	Restricted Parameter Space
Position	`x, y`	Positive Integers
Dimensions	`w, h`	Positive Integers
Rotation	`rotation`	Integer Degrees
Alignment	`align`	`start`, `middle`, `end`
Layering	`index`	Sequential Rank Index

C Details of Evaluation Settings

Implementation Details. We utilize the popular `excalidraw`[1] library as the underlying canvas engine. See Appendix B.4 for its schema description. The agentic framework is orchestrated using LANGGRAPH and powered by VLM backbones. For the design expertise memory, we use a vector store and embedding model to retrieve relevant expertise items by semantic similarity. Unless otherwise stated, generation uses a temperature of 0.7, one localized refinement pass, and top-k retrieval for each knowledge type.

Data Splits and Evolution Settings. CanvasBench defines a balanced held-out test set and a remaining pool for knowledge evolution. The main tables report a balanced evaluation subset spanning the diagram types in CanvasBench. For memory evolution, each round selects samples for iterative exploration with requirement variants, refinement traces, promotion thresholds, and maintenance operations. Detailed prompts are provided in our codebase[2].

Baseline Settings. The baselines cover four paradigms: *Pixel-based Generation*, where image-generation models directly produce raster diagrams; *Code-based Synthesis*, where LLMs generate LaTeX TikZ code [34]; *One-shot Schema Generation*, where LLMs directly generate the canvas schema manifest; and *Agentic Diagram Systems*, where existing multi-agent diagramming workflows are adapted to the same task format. For fair comparison, Table 1 separates method names from backbone models and evaluates non-pixel baselines under GPT-5.5, Gemini-3.1-Pro, DeepSeek-V4-Pro, and DeepSeek-V4-Flash whenever the method can use a backbone VLM.

C.1 Multi-Seed and Cross-Judge Checks

Generation variability. We repeated the main comparison with three independent generation seeds, using one representative model per paradigm and GPT-5.5 for the LLM-based methods. Table 6 reports the mean and standard deviation across seeds for each metric on the five-point scale. The standard deviations describe variability due to generation seeds. EvoDesign has higher mean scores than TikZ and one-shot prompting on all ten metrics; pixel generation has higher means on Visual Aesthetics, Space Balance, and Cognitive Ease.

[1] `https://excalidraw.com/`

[2] `https://github.com/GeminiLight/EvoDesign`

Table 6: Three-seed comparison: mean ± standard deviation across independent generation seeds. Pixel generation uses GPT-4o-Image; TikZ, prompting, and EvoDesign use GPT-5.5. The two blocks show all ten CanvasBench metrics.

Method	C_{CF}	C_{CL}	C_{SR}	V_{VA}	V_{SC}
Pixel	2.14 ± 0.17	2.31 ± 0.13	2.47 ± 0.11	3.77 ± 0.10	3.66 ± 0.08
TikZ	3.37 ± 0.09	3.45 ± 0.12	3.55 ± 0.07	2.71 ± 0.16	2.54 ± 0.13
Prompting	3.91 ± 0.11	3.53 ± 0.09	3.58 ± 0.12	2.50 ± 0.15	2.86 ± 0.12
EvoDesign	4.09 ± 0.08	4.07 ± 0.10	4.15 ± 0.04	3.58 ± 0.13	3.81 ± 0.05
Method	V_{FC}	V_{SB}	G_{CE}	G_{CA}	G_{SE}
Pixel	3.62 ± 0.14	3.60 ± 0.09	4.01 ± 0.10	2.32 ± 0.16	2.88 ± 0.12
TikZ	3.29 ± 0.15	2.89 ± 0.14	3.81 ± 0.12	3.01 ± 0.10	2.82 ± 0.15
Prompting	2.55 ± 0.18	2.03 ± 0.16	2.77 ± 0.13	3.00 ± 0.10	2.45 ± 0.14
EvoDesign	3.68 ± 0.11	3.56 ± 0.09	3.94 ± 0.10	3.48 ± 0.07	3.11 ± 0.12

Judge agreement. For a separate 100-item check, we compared Kimi-K3 and GPT-5.2 judgments using per-item Spearman rank correlations for each metric. Correlations range from 0.58 for Cognitive Ease to 0.74 for Content Fidelity, with a mean difference of 0.05 in the aggregate five-point score. The metric-wise correlations are 0.74, 0.63, 0.69, 0.70, 0.66, 0.61, 0.72, 0.58, 0.71, and 0.64 for C_{CF}, C_{CL}, C_{SR}, V_{VA}, V_{SC}, V_{FC}, V_{SB}, G_{CE}, G_{CA}, and G_{SE}, respectively.

D Details of CanvasBench Dataset and Benchmark

D.1 Description of Data Collection Pipeline

As illustrated in Figure 3, we implement a three-stage pipeline to filter and synthesize diagram-instruction pairs using diagram editability as the selection criterion.

(1) Structured Image Search. To ensure broad coverage of real-world scenarios, we construct a search matrix intersecting two axes. We manually curated 21 diagram types (e.g., Flowcharts, Mind Map) specifically tailored for canvas editability, while adopting the 30 vertical domains (e.g., Finance, Healthcare) from the established MMMU benchmark [40, 10] to guarantee content diversity. The full taxonomy is detailed in Appendix D.2. We utilize the Google Images API to retrieve candidate queries based on these enumerated tuples, i.e., "`diagram_type` diagram of `vertical_domain`".

(2) Diagram Editability Validation. We use Qwen3-VL-Max [1] to filter out hand-drawn sketches, complex data visualizations, photo infographics, and blurred documents, retaining diagrams whose structure can be represented as editable objects. Section 5.1 reports the independent annotation and deduplication protocol.

(3) Content & Intent Synthesis. Since raw web data lacks explicit user instructions and structured captions, we employ a data completion strategy to synthesize the missing information. We instruct Qwen3-VL-Max to jointly analyze both the diagram image and its associated webpage context to reverse-engineer generation. This process generates both a dense diagram description grounding the visual topology and a canonical user query (e.g., "Design a deployment pipeline...") representing the likely user intent. Consequently, this step transforms the raw data into (image, query, content) triplets.

D.2 Taxonomy for Diagram Image Retrieval

Table 7 lists the 21 diagram types and 30 vertical domains combined to construct the retrieval queries.

D.3 Further Dataset Analysis

We analyze description length and domain coverage to characterize CanvasBench.

Semantic Complexity and Information Density. Constructing high-fidelity diagrams requires processing not only topological structures but also dense textual information. We quantify this

Table 7: Detailed Taxonomy for Diagram Image Retrieval. The taxonomy intersects 21 structural types with 30 semantic domains.

Dual-axis Taxonomy for Diagram Image Retrieval

Axis 1: Structural Diversity (21 Diagram Types)

Selection criterion: Diagram editability.

- Architecture Diagram
- Class Diagram
- Concept Map
- Data Flow Diagram
- Entity Relationship (ER)
- Fishbone Diagram
- Flowchart
- Funnel Chart
- Gantt Chart
- Hybrid Diagram
- Matrix Chart
- Mind Map
- Org Chart
- Pyramid Chart
- Sequence Diagram
- State Diagram
- Swimlane Diagram
- SWOT Analysis
- Timeline
- Tree Diagram
- Venn Diagram

Axis 2: Semantic Diversity (30 Vertical Domains)

Taxonomy Source: Adopted from MMMU Benchmark [40].

Art & Design	Art, Design, Music, Art Theory
Business	Accounting, Economics, Finance, Management, Marketing
Science	Biology, Chemistry, Geography, Math, Physics
Health & Medicine	Basic Medicine Science, Clinical Medicine, Diagnostics, Pharmacy, Public Health
Humanities	History, Literature, Psychology, Sociology
Tech & Eng.	Agriculture, Architecture, CS, Electronics, Energy, Materials, Mechanical Engineering

complexity by analyzing the character count distribution across the 21 diagram types, as illustrated in Figure 7. The results demonstrate that CanvasBench maintains a high level of information density, with the median text length for most categories exceeding 1,000 characters. Notably, knowledge-intensive types such as *Concept Maps*, *Mind Maps*, and *Entity Relationship (ER)* diagrams exhibit substantial variance and higher upper quartiles, reflecting their role in organizing complex, multi-faceted concepts. Even structurally rigid types like *Sequence Diagrams* maintain significant textual volume, underscoring the necessity for models to possess strong long-context understanding capabilities alongside layout generation skills.

Domain Coverage. Figure 8 presents the vertical-domain coverage of the current CanvasBench corpus. The dataset follows the 30 MMMU-derived vertical domains used during retrieval, which provide broad coverage of terminology and logical conventions for knowledge evolution, while the held-out test set remains balanced by diagram type.

D.4 Examples of Diagram Data

To provide a concrete visualization of the diversity and quality of CanvasBench, we present several representative samples in Figure 9. These examples showcase the benchmark's coverage across various structural types, including state diagrams, SWOT analysis, flowcharts, and mind maps, and their corresponding vertical domains such as Computer Science, Architecture, Design, and Art. Each data point in the benchmark is a curated triplet consisting of a high-fidelity diagram image, a canonical user query, and a dense textual description. We can observe that

- *Structural Fidelity*: The examples demonstrate complex topological features, such as the cyclical transitions in the library asset lifecycle (State Diagram) and the hierarchical branching of artistic movements (Mind Map).

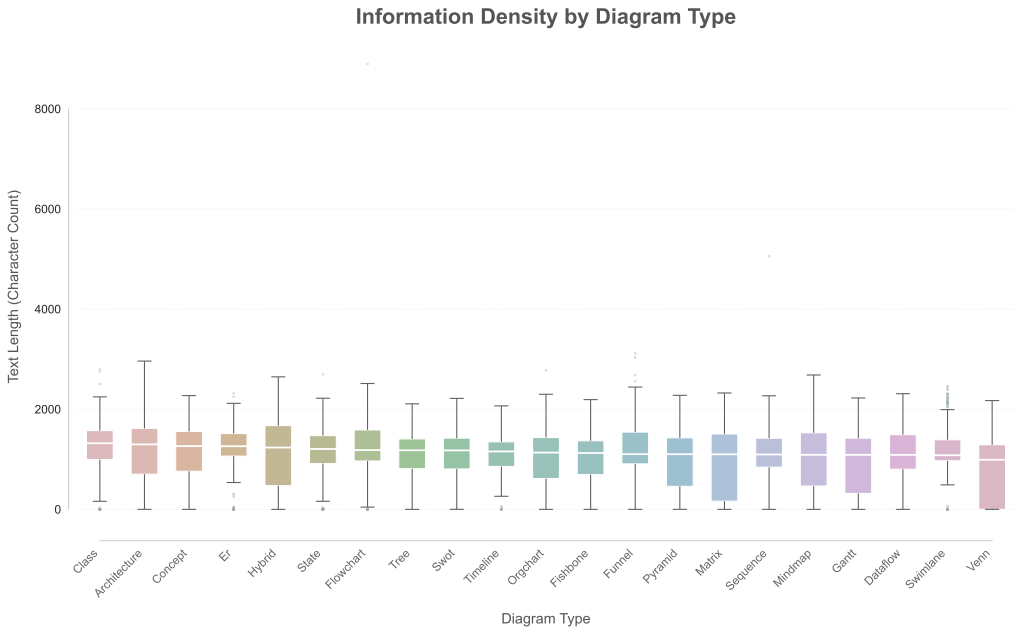


Figure 7: Distribution of description character counts by diagram type.

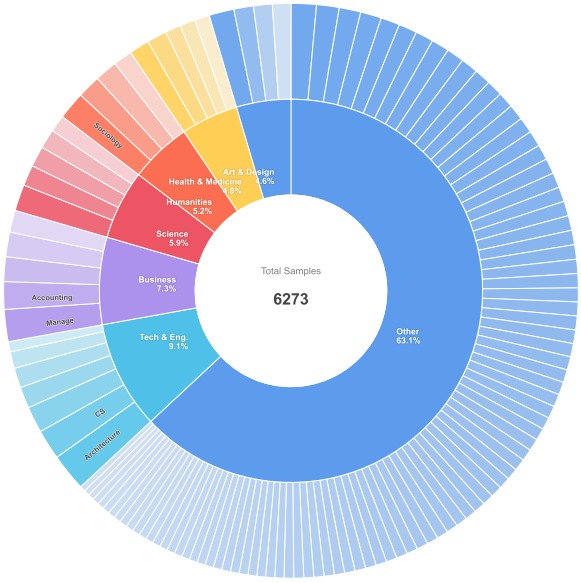


Figure 8: Vertical domain coverage in the current corpus. The chart summarizes the 30 MMMU-derived vertical domains used for CanvasBench construction.

- *Semantic Richness*: The associated source texts (right-hand panels) exhibit high information density, typically exceeding 1,000 characters. This requires models to not only extract entities but also to synthesize latent relationships into a coherent visual layout.

- *Diagram Editability*: The samples contain object-based structures suitable for editable nodes, edges, and groups.

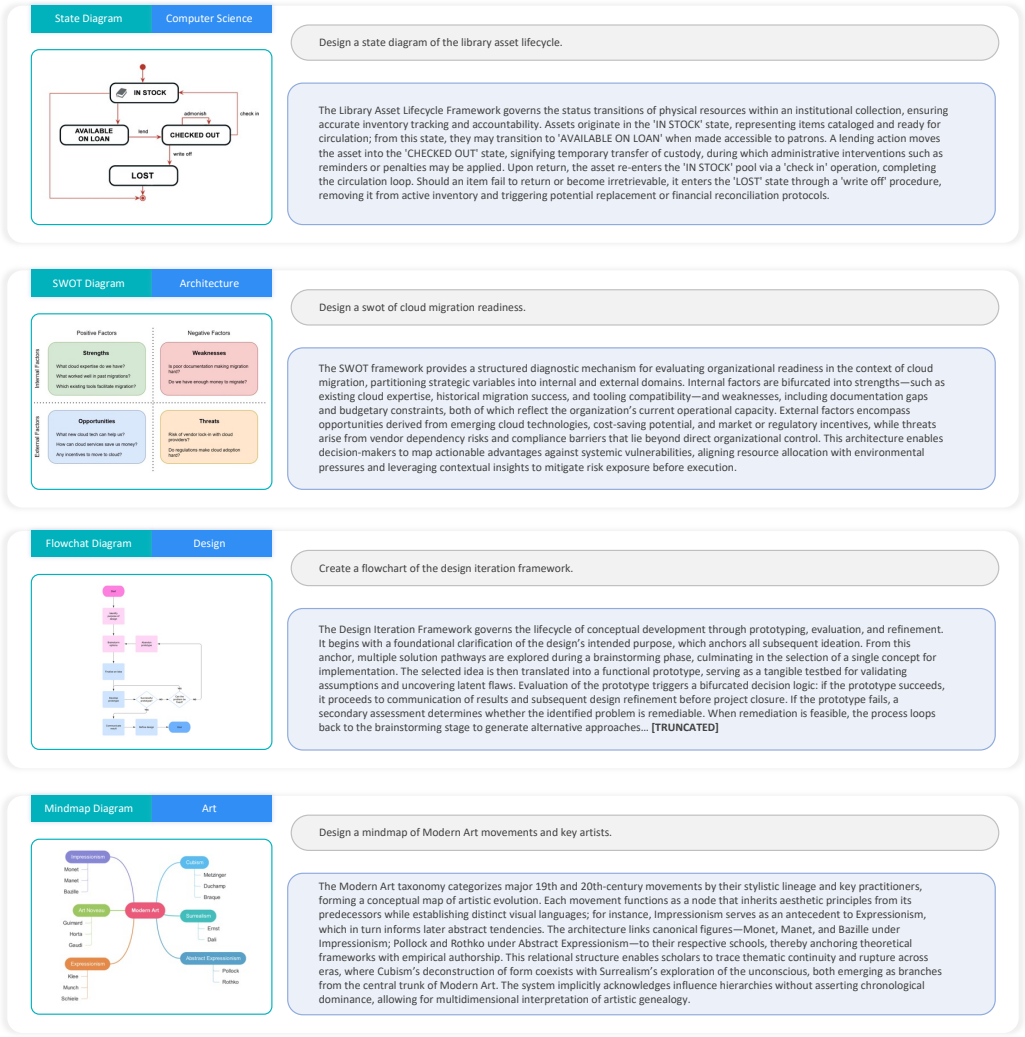


Figure 9: Data examples in CanvasBench.

D.5 Definitions and Prompts of Evaluation Metrics

In this section, we provide the detailed definitions and scoring rubrics for the evaluation framework. The evaluation utilizes VLM-based judges to assign scores on a 5-point Likert scale across three primary axes.

(a) Content Integrity Dimension ($\mathcal{C}$) Focuses on the semantic accuracy of the translation from source text to schema.

- *Content Fidelity* (C_{CF}): Verifies if every visual element is factually supported by the source text and penalizes "hallucinated" entities or relationships.
- *Concept Logic* (C_{CL}): Evaluates structural correctness, ensuring parent-child hierarchies and edge directions follow logical dependency rules.
- *Semantic Relevance* (C_{SR}): Assesses information density, ensuring key concepts are covered (Recall) and node text remains concise rather than verbose.

(b) Visual Presentation Dimension ($\mathcal{V}$) Assesses the rendering quality and adherence to professional design principles.

- *Visual Aesthetics* (V_{VA}): Rates the overall professional appearance, composition harmony, and effective use of negative space.

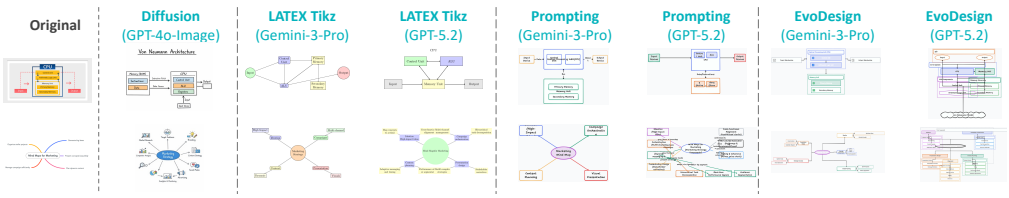


Figure 10: Representative diagram outputs from different generation paradigms on the same task.

- *Styling Consistency (V_{SC})*: Checks for a uniform "Design Language," ensuring font hierarchy, color themes, and stroke styles are consistent across similar element types.

- *Flow Coherence (V_{FC})*: Assesses the clarity of the reading path and penalizes ambiguous or "spaghetti" edge routing.

- *Space Balance (V_{SB})*: Evaluates if layout density is balanced across the canvas and identifies spatial conflicts such as overlaps.

(c) Cognitive Utility Dimension ($\mathcal{G}$) Section 5.3 defines the three cognitive metrics: Cognitive Ease (G_{CE}), Comprehension Accuracy (G_{CA}), and Self-contained Explanation (G_{SE}). Table 8 lists their focus and scoring sub-items.

Table 8: Overview of Evaluation Metrics

Dimension	Metric Name	Symbol	Focus	Sub-items
Content Integrity	Content Fidelity	C_{CF}	Trustworthiness & Fact-checking	Factual Consistency, Hallucination Rate, Evidence Traceability
	Concept Logic	C_{CL}	Logical Structure & Relationships	Edge Correctness, Relation Type Accuracy, Self-Consistency, Hierarchy Rationality
	Semantic Relevance	C_{SR}	Information Density & Alignment	Coverage, Conciseness, Context Alignment
Visual Presentation	Visual Aesthetics	V_{VA}	Overall Look & Feel	Composition & Whitespace, Harmony, Professionalism
	Styling Consistency	V_{SC}	Standardization & Uniformity	Font Consistency, Line & Shape Consistency, Color Theme Consistency
	Flow Coherence	V_{FC}	Reading Path & Navigation	Main Reading Direction, Low Path Ambiguity, Key Path Trackability
	Space Balance	V_{SB}	Layout & Density	Spatial Equilibrium, Density, Overlap & Crossing
Cognitive Utility	Cognitive Ease	G_{CE}	Mental Effort & Friction	Subjective Ease, Reading Friction, Mental Load
	Comprehension Accuracy	G_{CA}	Understanding & QA Correctness	Factual QA, Structural QA, Interpretation Consistency
	Self-contained Explanation	G_{SE}	Independence from Source	Self-containment, Narrative Completeness, Misinterpretation Risk

E Qualitative Artifacts and Knowledge Examples

E.1 Generated Diagram Artifacts

E.2 Geometric Refinement Example

Verification identifies two edge-node intersections: the Distill-to-Memory edge traverses Critique and Revise. The refiner routes the memory return along the lower perimeter and separates the retry connector from the outgoing trace. All nine nodes stay in place, and their labels and eleven directed connections are preserved. Figure 11 shows the complete refined workflow; Figure 5 pairs the initial defect detail with the full result.

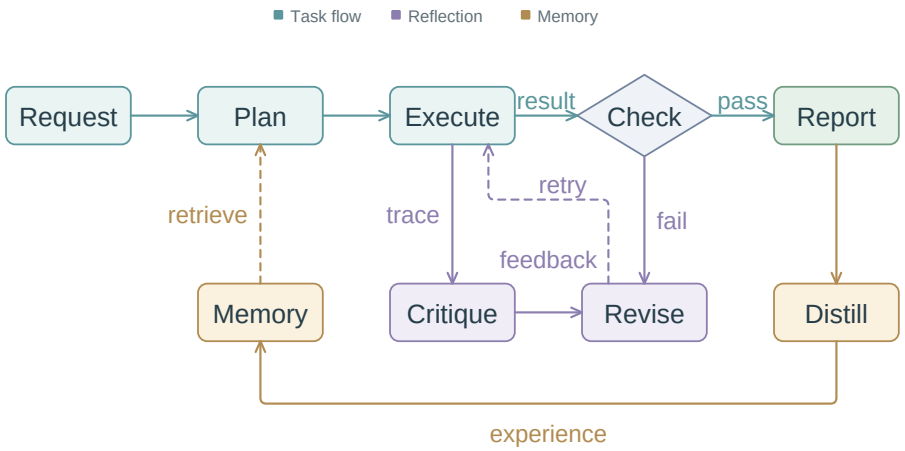


Figure 11: Refined research-assistant workflow. The task path, reflection loop, and memory return occupy separate routes while retaining all node positions and semantic connections.

E.3 Examples of Knowledge Items

Tables 9, 10, 11 show example entries in the three-tier design memory $\mathcal{M} = \{\mathcal{K}^s, \mathcal{K}^g, \mathcal{K}^p\}$, including their content, tags, and application conditions.

Table 9: Example schema for sample strategies (K^s) within the design memory $\mathcal{M}$.

Refiner 01

ReAct Algorithm workflow trace

History • Flowchart

Description

Preserves a localized refinement record with source findings, applied edits, and post-edit diagnostics.

Content

Reasoning paths and feedback loops can be represented as explicit nodes and transitions when supported by the source content.

When to use

When depicting reasoning steps, actions, and feedback loops in a flowchart.

Structure 02

Microservices decomposition trace

CS • Architecture

Description

Records the parsing of system descriptions into discrete service entities.

Content

High-entropy source text is filtered into a unique entity pool to reduce identity fragmentation in the diagram.

When to use

When resolving topological complexity in system architecture graphs.

Table 10: Example schema for domain guidelines (K^g) within the design memory $\mathcal{M}$.

Layout History • Flowchart 01

Node icon positioning

Description

Spatial placement rule for protecting text clarity.

Content

Place auxiliary icons outside dense text regions when the canvas has sufficient nearby whitespace.

When to use

When adding icons to text-dense history flowcharts.

Styling Finance • Gantt 02

Time-series alignment

Description

Alignment rule for financial temporal data.

Content

Align time-series data horizontally and distinguish fiscal periods with consistent visual encodings.

When to use

When displaying multiple fiscal periods in financial timelines.

Table 11: Example schema for general principles (K^p) within the design memory $\mathcal{M}$.

Layout Universal • Universal 01

Spatial exclusion for semantic clarity

Description

Domain-agnostic rule distilled from supported lower-tier entries.

Content

Auxiliary elements should be positioned in available negative space adjacent to container boundaries to preserve information integrity.

When to use

When a diagram contains optional annotations that may otherwise collide with core content.

Styling Universal • Universal 02

Functional modularity

Description

Cross-domain pattern emphasizing clear cluster separation.

Content

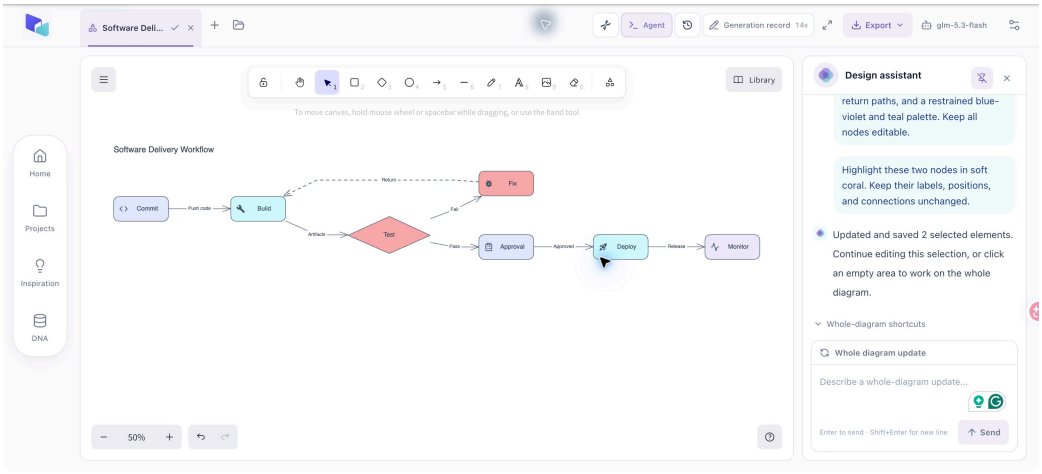
Visual hierarchy should map dominant attributes to critical entities while mapping peripheral nodes to muted styles.

When to use

When establishing global consistency across chart types or styles.

F End-to-End User Study

F.1 Human-Centered Web Application



Current browser application: editable canvas and natural-language local refinement.

Figure 12: EvoDesign browser application. The workspace combines project and DNA navigation, an editable `excalidraw` canvas, export controls, and a design assistant. The conversation records a local request that updates two selected nodes while preserving the rest of the workflow.

Figure 12 shows the project and design-memory navigation alongside the editable canvas and export controls. The right-side design assistant records a selection-scoped request that updates two node fills without changing labels, positions, or connections.

F.2 Human Study Setup

We conducted a user study with 36 participants to evaluate EvoDesign in the two human-facing parts of the workflow that are not fully captured by automatic metrics: personalized design memory and editable refinement. All participants had prior experience in diagram authoring. They used the web application for their own diagramming tasks at least ten times within a day before completing the questionnaire. The questionnaire asked about personalized memory, which learns design preferences from lightweight feedback, and editable refinement through the canvas schema. Each item was rated on a five-point Likert scale from strongly disagree to strongly agree.

The study examined two groups of questions. *Personalized memory* asked whether participants perceived their own memory as better than generic memory, reported fewer edits, found the outputs readable, and considered their preferences captured. *Editable refinement* asked about perceived refinement speed, edit quality, workflow preference, and preservation of source meaning. This setup follows the central use case of EvoDesign: generating an initial editable diagram, adapting it to user-specific design preferences, and supporting controlled refinement instead of forcing users to regenerate a static artifact.

Figure 6 in the main paper reports all eight item means and standard deviations.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction identify editable canvas generation, verification-guided design expertise evolution, and CanvasBench as the contributions. Sections 4–6 present the method and evaluation, with scope limitations in Appendix A.1.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: Appendix A.1 discusses canvas-schema expressivity, cold-start behavior, cross-model transfer, reference-derived benchmark inputs, and computational cost.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

   Justification: The paper presents no theoretical theorems or proof-based results. Section 3 defines the task and representation, and Appendix B.1 provides procedural pseudocode.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: Section 4 and Appendix B.1 specify the generation and evolution algorithms, score conversion, mixture weights, and promotion gates. Section 6 and Appendices C and D report the settings, construction procedure, metrics, and robustness checks.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No].

   Justification: The public repository provides code, prompts, baseline adapters, evaluation tools, and run instructions at `https://github.com/GeminiLight/EvoDesign`. CanvasBench training and test data are not included in the current public release.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: Section 6 and Appendix C report backbone models, decoding settings, refinement passes, splits, and baselines. Section 4 specifies the 0.6/0.4 verification weights, $\tau = 0.1$, held-out selection, and promotion evidence requirements; Appendix C.1 reports the three-seed and cross-judge setups.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: Appendix C.1 reports mean $\pm$ standard deviation over three independent generation seeds and a 100-item cross-judge check. Figure 6 reports questionnaire means and one standard deviation for the completed 36-participant study.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No].

   Justification: Appendix A.1 discusses the multiple model calls and interactive latency, but full hardware specifications and per-experiment compute budgets are not reported.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: To the best of our knowledge, the research conforms to the NeurIPS Code of Ethics. Appendix A discusses misleading outputs and responsible handling of web-derived visual materials.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: Appendix A discusses accessible diagram creation and human-AI co-creation, alongside misleading diagrams from incomplete source content and attribution, licensing, and privacy concerns for web-derived materials.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: The released asset is general-purpose diagramming software, and the benchmark consists of structured diagrams and instructions. The work does not release a new pretrained model or a high-risk personal-data resource.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No].

    Justification: Models, libraries, and benchmarks are credited, and the public repository provides software and third-party dependency notices. The manuscript does not establish per-image licenses or redistribution permissions for every web-derived source image.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: Section 5 and Appendix D document CanvasBench collection, taxonomy, examples, and evaluation metrics. The public repository documents the implementation and its usage.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [No].

    Justification: The 36-participant study is complete: Section 6.6, Figure 6, and Appendix F provide the usage procedure, interface, and all eight questionnaire summaries. The full participant instructions and compensation arrangements are not reported.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [No].

    Justification: The manuscript reports the completed 36-participant study. Its institutional review or exemption status, consent procedure, and participant risk disclosures are not reported.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes].

    Justification: Sections 4–6 and Appendix B.1 describe LLM/VLM use in generation, critique, refinement, expertise distillation, benchmark synthesis, and judging. Section 6 reports backbone models and decoding settings.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.