CRAFTER: A MULTI-AGENT HARNESS FOR EDITABLE SCIENTIFIC FIGURE GENERATION FROM DIVERSE INPUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Scientific figures are among the most effective means of communicating complex research ideas, yet producing publication-quality illustrations remains one of the most labor-intensive parts of paper preparation. Existing automated systems each target a single figure type under text-only input, leaving the diversity of types and conditions researchers actually use unaddressed; their raster outputs further cannot be locally revised. Because scientific figures are structured compositions of discrete semantic components, the localized errors that generators produce on such layouts demand not a stronger backbone but a *harness*. We instantiate this harness in two complementary systems: CRAFTER, a multi-agent harness for figure generation that generalizes across figure types and input conditions without architectural changes, and CRAFTEDITOR, which applies the same pattern to convert raster outputs into editable SVGs. We also introduce CRAFTBENCH, a benchmark spanning three figure types and four input conditions with human quality annotation. Experiments show that CRAFTER substantially outperforms both standalone generators and agentic baselines on PaperBanana-Bench and CRAFTBENCH, with ablations confirming each component's independent contribution, and CRAFTEDITOR converts CRAFTER's outputs into editable SVGs more faithfully than prior methods.

1 INTRODUCTION

Text-to-image generation has advanced rapidly, with recent models producing photorealistic and design-quality images across creative, medical, and scientific domains (Wu et al., 2025; Z.ai, 2026; Black Forest Labs, 2024). One area where this progress has yet to translate into practical tools is scientific illustration, where producing publication-quality figures remains one of the most labor-intensive parts of paper preparation. Recent work tackles this from two directions: agentic pipelines that pair planning agents with powerful image generators to produce figures from text (Zhu et al., 2026a;b; Sun et al., 2025; Guo et al., 2025; Kukreja et al., 2026; Yang et al., 2026), and code-generation methods that synthesize editable diagrams in TikZ or similar formats (Belouadi et al., 2024; Zala et al., 2024; Greisinger & Eger, 2026; Zheng et al., 2025). While encouraging, these approaches fall short of real-world demands in two respects.

First, existing systems are narrow in scope. In practice, researchers produce figures across a spectrum of types, from academic diagrams to posters and infographics, and rarely begin from text alone; instead, they iterate from rough sketches, partial layouts, or reference visual elements or icons. Current methods, by contrast, focus predominantly on text-to-image generation (Zhu et al., 2026a;b; Guo et al., 2025; Sun et al., 2025), leaving the diversity of figure types and input conditions entirely unaddressed. AutoFigure extends the input to long-form text from papers, surveys, blogs, and textbooks, but still accepts only text input. Existing evaluations reflect the same narrow scope, covering only text-to-image generation of methodology figures (Zhu et al., 2026a) with no mechanism to assess whether a system generalizes across figure types or preserves a user's conditioning input. Second, output images are not practically editable. Raster-based generators produce static images that cannot be locally revised, a problem when researchers need to revise individual labels, swap color schemes, or rearrange components. Code-generation methods yield editable output but lack the visual richness of icons and stylized layouts; recent raster-to-vector attempts (Lin et al., 2026; BIT-DataLab Contributors, 2026)

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

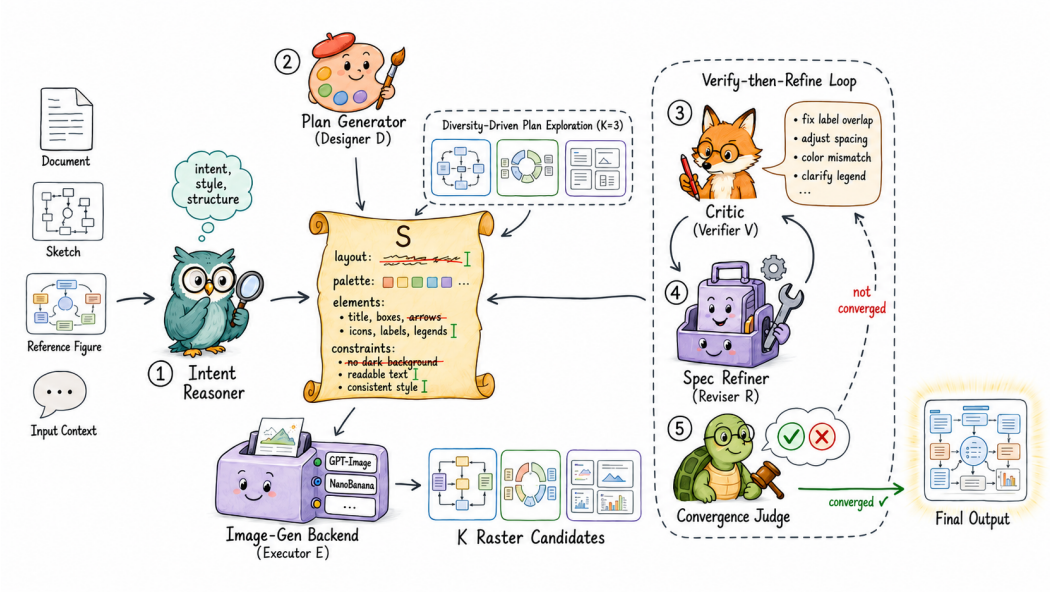


Figure 1: CRAFTER architecture. A four-role harness loops over the shared evolving specification $\mathcal{S}$, with a convergence judge routing each round to accept, refine, or revert. (Generated by CRAFTER.)

remain limited by unreliable element extraction and fragile composition. A complete scientific figure pipeline must therefore extend beyond generation to structurally editable output.

Addressing the generation challenge requires more than a stronger backbone. Scientific figures, unlike natural images, are structured compositions of discrete semantic components: labeled boxes, directional arrows, icons, and annotations, each carrying specific meaning within precise spatial relationships. Modern generators exhibit high output variance on such structured layouts, producing localized errors such as garbled labels and misaligned connectors that rewording prompts alone cannot fix. Naive retry is ineffective because each attempt produces a different constellation of failures, and accumulating free-text corrections across iterations introduces contradictions that further degrade quality. The same pattern holds for raster-to-vector conversion, where imprecise extraction and fragile composition persist across one-shot attempts. What is needed in both settings is not a better generator but a *harness* (Young, 2025; Pan et al., 2026; Bui, 2026; Si et al., 2026a): an orchestration layer that wraps an existing engine with an evolving structured specification as its memory, enabling targeted correction of individual failure points and closed-loop verification against the original intent.

We instantiate this harness in two complementary systems. **CRAFTER** (Figure 1) is a multi-agent harness for scientific figure generation in which five cooperating agents (an intent reasoner, a plan generator, a critic, a specification refiner, and a convergence judge) share an evolving figure specification as the pipeline's structured memory, while an image-generation backend handles all rendering. Three mechanisms underpin the design: *diversity-driven plan exploration* generates multiple candidate framings in parallel; a *structured corrective layer* accumulates critique-driven typed edits into the shared specification, preventing the prompt contradictions that plague free-text revision; and a *verify-then-refine loop* iterates with a directive critic that issues targeted corrections rather than scalar scores. Because all task-specific behavior resides in agent prompts, the same architecture generalizes across figure types and input conditions without structural change. **CRAFTEDITOR** applies the same harness pattern to convert raster figures into editable SVGs through three sequential phases. An extraction phase strips text overlays and visual clutter to obtain clean graphical assets; a processing phase captions each asset and classifies it as vector or raster; and a composition phase assembles these assets into an SVG skeleton and iteratively refines the result via a hybrid critic.

To evaluate across this broader scope, we introduce **CRAFTBENCH**, a 279-sample benchmark spanning three figure types and four input conditions, curated from published papers across 18 subject areas, award-tier conference posters, and research blogs through a multi-stage pipeline with human quality annotation. Following previous work (Zhu et al., 2026a), we adopt an evaluation protocol that

assesses output quality against human-drawn targets using VLMs as judges. On PaperBanana-Bench and CRAFTBENCH, CRAFTER substantially outperforms both standalone generators and the strongest agentic baseline under controlled comparison. Ablations validate each mechanism: removing any single component causes a 5.04 to 8.90 point drop. CRAFTEDITOR converts generated outputs into editable SVGs, outperforming all baselines on a three-VLM ensemble evaluation, making our method a pioneering step toward the full generation-to-editing workflow.

2 RELATED WORK

Scientific figure creation. Automated scientific figure creation falls into two families. Code-generation methods synthesize editable diagrams in code such as TikZ from text descriptions (Belouadi et al., 2024; Zala et al., 2024; Greisinger & Eger, 2026), but are restricted to schematic diagrams and lack the visual richness of icons and stylized layouts. Agentic pipelines pair LLM agents with image generators to produce high-quality raster figures for methodology plots (Zhu et al., 2026a;b), posters (Sun et al., 2025), architecture diagrams (Guo et al., 2025), and educational illustrations (Kukreja et al., 2026; Yang et al., 2026), yet their raster outputs cannot be easily revised. Recent efforts to bridge raster output and editability remain preliminary: AutoFigure-Edit (Lin et al., 2026) detects elements and emits an SVG in a single LLM call, and Edit-Banana (BIT-DataLab Contributors, 2026) converts segmentation and OCR outputs into DrawIO cells. Both families share a limitation: each system targets a single figure type, accepts only text input, and ignores the diversity of conditions in real research workflows. Work on agent orchestration (Young, 2025; Pan et al., 2026) and iterative self-correction (Madaan et al., 2023) has shown that the harness layer matters as much as the underlying model, but this principle remains unexplored for scientific figures.

Benchmarks and evaluation. Evaluation methods for scientific figure generation remain as narrow as the systems they measure. PaperBanana-Bench (Zhu et al., 2026a) and Paper2SysArch (Guo et al., 2025) evaluate only text-to-image generation of academic diagrams. SridBench (Chang et al., 2025) covers thirteen fields but remains limited to text-to-image generation without conditional inputs. IGenBench (Tang et al., 2026) targets text-to-infographic reliability with a decomposed verification framework but covers only infographics. SciFlow-Bench (Zhang et al., 2026) inverse-parses generated diagrams into structured graphs to measure structural recoverability, but is limited to flowchart diagrams. None of these benchmarks tests cross-type, cross-condition generalization. CRAFTBENCH fills this gap with coverage of three figure types and four input conditions.

3 METHOD

Generating scientific figures reliably faces three technical difficulties: high output variance on complex structured layouts, prompt degradation from accumulated free-text corrections, and the absence of structured corrective feedback. These difficulties call for a harness layer that orchestrates planning, verification, and revision around the generator rather than improving it (§3.1). We equip the harness with three targeted mechanisms and instantiate them in CRAFTER (§3.2); CRAFTEDITOR reuses the same pattern for raster-to-vector conversion (§3.3).

3.1 THE HARNESS ABSTRACTION

A *harness* is an orchestration layer that wraps an executor with planning, verification, and structured revision, detecting and correcting the executor's failure modes without modifying the executor itself. In our setting, the executor is an image generator (for CRAFTER) or an image editor and element-injection code (for CRAFTEDITOR). We formalize the harness as a four-role loop over a shared *evolving specification* $\mathcal{S}$, a structured record that accumulates the current plan, the revision history, and prior diagnostics (Figure 1). At each round t:

$$p_t = \mathcal{D}(\text{input},\ \mathcal{S}_{t-1}), \qquad a_t = \mathcal{E}(p_t), \tag{1}$$

$$d_t = \mathcal{V}(a_t,\ \text{input},\ \mathcal{S}_{t-1}), \qquad \mathcal{S}_t = \mathcal{R}(d_t,\ \mathcal{S}_{t-1}), \tag{2}$$

where the designer $\mathcal{D}$ produces an actionable plan p_t, the executor $\mathcal{E}$ renders it into an artifact a_t, the verifier $\mathcal{V}$ emits a *directive diagnostic* d_t (per-dimension scores, identified defects, and suggested corrections, as opposed to a scalar quality score), and the reviser $\mathcal{R}$ applies *typed edits* to $\mathcal{S}_{t-1}$, each

Table 1: Harness role assignments for CRAFTER and CRAFTEDITOR.

Role	CRAFTER	CRAFTEDITOR
$\mathcal{D}$ Designer	Plan generator	SVG skeleton generator
$\mathcal{E}$ Executor	Image-generation backend	Element-injection code
$\mathcal{V}$ Verifier	Multi-dim. critic	Hybrid critic
$\mathcal{R}$ Reviser	Specification refiner	SVG editor

a structured operation (adding a layout constraint, banning an artifact category, resizing a named element) that modifies the specification in place rather than appending free text to the prompt. The loop terminates when $\mathcal{V}$ accepts a_t or a round budget T is reached, returning $a^* = \arg\max_\tau \text{score}(d_\tau)$.

Two properties make this loop effective for scientific figures: $\mathcal{E}$ is pluggable, so all task-specific behavior resides in the prompts of $\mathcal{D}$, $\mathcal{V}$, and $\mathcal{R}$; and $\mathcal{R}$ writes typed edits to a shared record rather than free-text additions to the prompt, keeping the specification internally consistent across rounds. Table 1 summarizes how CRAFTER and CRAFTEDITOR instantiate each role.

3.2 CRAFTER: HARNESS FOR FIGURE GENERATION

Given a context $\mathbf{c}$ (e.g., papers, reference images, or sketches) and an instruction $\mathbf{q}$, the CRAFTER harness produces a publication-quality raster figure a^* and the final specification $\mathcal{S}_T$.

Five agents implement the four harness roles (Figure 1; Table 1). An *intent reasoner* infers the figure's communicative role and required visual elements from $(\mathbf{c}, \mathbf{q})$, seeding the initial specification $\mathcal{S}_0$. The plan generator $\mathcal{D}$ proposes candidate visual plans from $\mathcal{S}_0$; the image-generation backend $\mathcal{E}$ renders each plan into a raster; the critic $\mathcal{V}$ evaluates every candidate against $\mathcal{S}$ and $(\mathbf{c}, \mathbf{q})$; and the specification refiner $\mathcal{R}$ writes typed edits back into $\mathcal{S}$. At each round, a convergence judge decides whether to accept, continue refining, or revert to a^*. Three mechanisms below address the failure modes identified in Section 1; implementation details are in Appendix C.

3.2.1 DIVERSITY-DRIVEN PLAN EXPLORATION

Image generators exhibit high inter-sample variance on complex, structured figures: qualitatively different layouts and compositions emerge across random seeds for a fixed prompt. A single-draw pipeline cannot recover from a structurally unsuitable sample, and hardcoding a style needlessly restricts the generation space. CRAFTER treats this variance as a search problem: $\mathcal{D}$ reads $\mathcal{S}_0$ and proposes K intent-conditioned candidate plans, each specifying a distinct visual framing (e.g., banner layout or multi-column grid). $\mathcal{E}$ renders all K plans in parallel, and the convergence judge selects the best candidate $a^{(1)} = \arg\max_k \text{score}(\mathcal{V}(a_k))$ as the starting point for refinement. $K \in \{1, 2, 3\}$ adapts to input complexity. Unlike extra refinement rounds, plan-level branching can escape a fundamentally unsuitable composition before any rendering budget is spent refining it.

3.2.2 STRUCTURED CORRECTIVE LAYER

Iterative repair via free-text revision instructions degrades rapidly: successive natural-language addenda introduce conflicting directives (e.g., "enlarge the title" followed by "reduce white space"), the generator absorbs the contradictions silently, and faithfulness deteriorates without any single round appearing anomalous. The structured corrective layer replaces free-text accumulation with *typed edits* on $\mathcal{S}$: at each round t, $\mathcal{R}$ converts the diagnostic d_t into structured operations $\{e_i\} = \mathcal{R}(d_t, \mathcal{S}_{t-1})$, where each e_i is one of the in-place operations of §3.1. The next round's prompt is assembled from this coherent record $\mathcal{S}_t$ rather than a growing stack of amendments.

3.2.3 VERIFY-THEN-REFINE WITH A DIRECTIVE CRITIC

Even with a well-chosen plan and an accumulating specification, first-generation outputs typically contain localized errors, such as missing components or duplicated regions. This mechanism comprises two components, a *directive critic* that diagnoses errors and a *verify-then-refine loop* that applies the corrections iteratively, each validated independently.

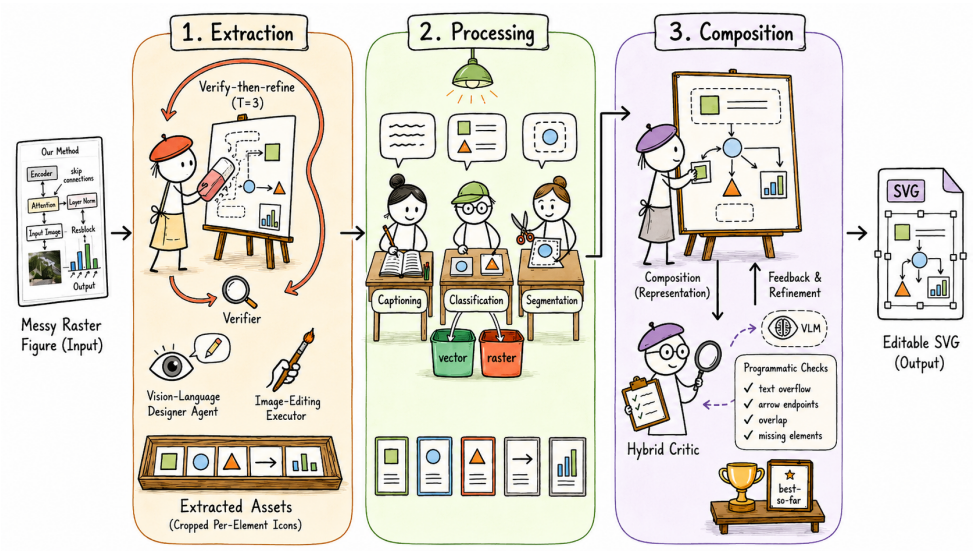


Figure 2: CRAFTEDITOR architecture. The same four-role harness drives the extraction and composition phases of a three-phase pipeline that converts a raster a^* into an editable SVG $\mathbf{v}$, with a hybrid critic refining the result each round. (Generated by CRAFTER.)

Directive critic. A scalar score (e.g., "5/10") provides no actionable target for the next round. $\mathcal{V}$ instead emits a directive diagnostic $d_t = \mathcal{V}(a_t, \mathbf{c}, \mathbf{q}, \mathcal{S}_{t-1})$ containing per-dimension scores along six axes, identified defects, suggested corrections, and a revised figure description. $\mathcal{R}$ converts d_t into edits on $\mathcal{S}$ (§3.2.2), and the prompt builder injects the corrections as fix-guidance into the next round.

Refinement loop. An *early-exit gate* bypasses the loop when the first-round output already satisfies acceptance thresholds on critical dimensions. Otherwise the loop runs for up to $T{=}3$ rounds, with a *best-so-far checkpoint* that reverts to a^* whenever the current round regresses, since language-model-driven iterative editing is empirically non-monotonic (Madaan et al., 2023).

3.3 CRAFTEDITOR: HARNESS FOR RASTER-TO-VECTOR CONVERSION

Raster figures do not support the element-level edits that research workflows demand (e.g., swapping icons or completing a partial diagram). CRAFTEDITOR converts a raster figure a^*, whether produced by CRAFTER or obtained externally, into a coordinate-faithful editable SVG $\mathbf{v}$ by instantiating the same harness loop (Eqs. 1 and 2) on vector composition rather than pixel synthesis.

Three phases organize the harness (Figure 2; Table 1). An *extraction phase* strips visual clutter from a^* and isolates per-element assets. A *processing phase* captions, grounds, and classifies each element (Appendix D). A *composition phase* assembles the assets into a final SVG and refines it through a critic-driven loop. The extraction and composition phases each instantiate the four-role harness: $\mathcal{D}$ authors a plan, $\mathcal{E}$ executes it, $\mathcal{V}$ inspects the result, and $\mathcal{R}$ revises the plan based on d_t.

Extraction: instruction-driven canvas cleaning. Scientific figures, particularly conference posters containing 25 to 50 visual assets, exhibit overlapping elements, text, and heterogeneous backgrounds that defeat off-the-shelf segmentation, which struggles to produce reliable boundaries and distinguish semantically relevant components on such cluttered layouts. CRAFTEDITOR replaces segmentation with an instruction-driven extraction loop. $\mathcal{D}$ (a vision-language agent) inspects a^* and authors a per-figure keep/delete plan p_t specifying which elements to preserve and which to remove. $\mathcal{E}$ (an instructable image editor) executes the plan at the pixel level, producing a cleaned canvas a_t. $\mathcal{V}$ inspects a_t and either accepts it or returns a diagnostic d_t that triggers another round, for at most $T{=}3$ iterations. Per-element assets are then cropped from the clean canvas, with a hallucination filter discarding blank, mismatched, or text-only extractions before they reach the composition phase.

Composition: iterative SVG assembly. A single language-model call to produce an SVG from the element inventory routinely generates layouts whose grid topology, arrow endpoints, or text labels

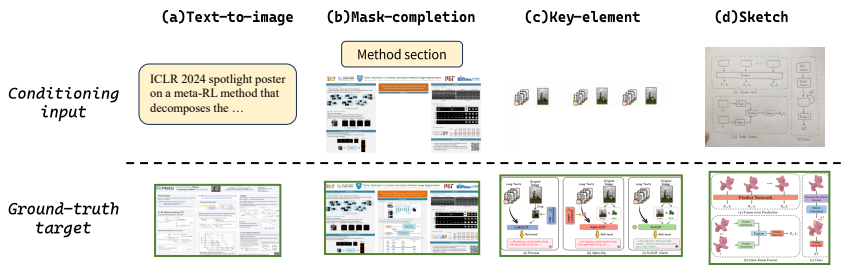


Figure 3: Representative CRAFTBENCH samples. Each column shows one task.

disagree with the input raster. The composition phase replaces this one-shot call with the full harness loop. $\mathcal{D}$ generates two candidate SVG skeletons at different decoding temperatures; the convergence judge selects the better candidate via a rapid visual comparison. $\mathcal{E}$ splices the extracted assets into the placeholders of the selected skeleton. $\mathcal{V}$ then evaluates the rendered SVG against a^* via a *hybrid critic* that combines two complementary channels: a vision-language model assessing global layout fidelity and semantic correspondence, and programmatic checkers auditing structural properties (text overflow, arrow-endpoint accuracy, element overlap, missing components) that vision-language evaluation alone tends to miss. $\mathcal{R}$ modifies the SVG source in response to d_t. The loop runs for up to $T{=}4$ rounds, with best-so-far reversion to the highest-scoring SVG guarding against non-monotonic regressions. As in CRAFTER, all task-specific behavior resides in prompts (Appendix D).

4 CRAFTBENCH

CRAFTBENCH spans three scientific figure types and four input conditions (text-to-image and three reference-conditioned tasks: mask-completion, key-element composition, and sketch-conditioned generation). Figure 3 shows representative inputs and ground-truth targets per task.

4.1 DATA CONSTRUCTION

Samples are built by a three-stage pipeline (full details in Appendix B). *Collection* draws academic figures from arXiv preprints across 18 subject areas, posters from award-tier conference papers, and infographics from long-form research blogs (Si et al., 2025). *Filtering* applies vision-language content classification, complexity scoring, and claim-alignment verification, leaving 553 candidates that human curation reduces to the final 279 samples spanning all four tasks and three styles. *Annotation* pairs each text-to-image sample with its caption and source paper-text, and constructs a conditioning input for the three reference-conditioned tasks. Every reference-conditioned sample is reviewed by three graduate-level annotators through a per-task interface and accepted only on unanimous agreement, with disagreements triggering revision until consensus.

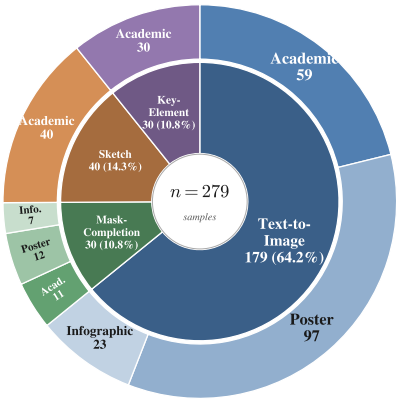


Figure 4: CRAFTBENCH distribution. Inner: task types; outer: per-task style.

4.2 STATISTICS

CRAFTBENCH contains 279 samples spanning four tasks and three styles (Figure 4). Text-to-image accounts for nearly two-thirds of the benchmark ($n{=}179$), with the three reference-conditioned tasks contributing mask-completion (30), sketch-conditioned generation (40), and key-element composition (30). Text-to-image and mask-completion span all three style families, while the sketch and key-element tasks are drawn entirely from academic figures. Academic figures form the largest share (140), complemented by 109 posters and 30 infographics.

Table 2: Results (%) on PaperBanana-Bench and CRAFTBENCH. **Bold** marks column-best; Δ is the gap between CRAFTER and its standalone generator. *On CRAFTBENCH, GPT-Image-2 returned valid outputs for only 260 of 279 inputs, likely due to instability and content-safety refusals. Differences are computed before rounding.

Method	PaperBanana-Bench					CRAFTBENCH				
	Faith.	Conc.	Read.	Aesth.	Overall	T2I	Mask	Sketch	KeyEl.	Overall
Standalone generators (open-source)										
GLM-Image	0.00	0.69	0.00	1.55	0.00	0.00	0.00	0.00	0.00	0.00
Qwen-Image	0.00	0.00	0.00	1.71	0.00	0.00	0.00	2.50	0.00	0.36
Standalone generators (closed-source)										
GPT-Image-2*	8.42	3.97	1.72	40.72	1.37	5.31	18.33	45.00	36.67	15.77
Nano Banana 2	15.07	11.99	26.88	47.95	11.13	8.38	30.00	61.25	23.33	19.89
Nano Banana Pro	25.43	36.13	37.16	50.52	22.43	17.32	21.67	46.25	21.67	22.40
Agentic frameworks										
AutoFigure (w/ Nano Banana 2)	12.33	6.85	3.26	7.71	1.37	1.40	0.00	8.75	0.00	2.15
PaperBanana (w/ Nano Banana 2)	28.10	52.41	42.64	61.68	33.73	18.72	36.67	60.00	31.67	27.96
PaperBanana (w/ Nano Banana Pro)	31.34	52.74	45.03	61.30	35.96	26.82	23.33	45.00	26.67	29.03
CRAFTER (w/ Nano Banana 2)	**38.18**	53.42	47.77	64.21	**50.34**	48.32	**45.00**	70.00	**40.00**	50.18
Δ vs. Nano Banana 2	+23.12	+41.44	+20.89	+16.27	+39.21	+39.94	+15.00	+8.75	+16.67	+30.29
CRAFTER (w/ Nano Banana Pro)	37.72	**55.31**	**48.46**	**65.29**	50.00	**52.51**	41.67	**73.75**	33.33	**52.33**
Δ vs. Nano Banana Pro	+12.29	+19.18	+11.30	+14.77	+27.57	+35.20	+20.00	+27.50	+11.67	+29.93

4.3 EVALUATION PROTOCOL

Our evaluation keeps the referenced VLM-as-judge philosophy of Zhu et al. (2026a), scoring each output against the human-drawn target and reporting a lenient win-rate, but redesigns the judge for the cross-type, cross-condition setting. A Gemini 3.5 Flash (Google DeepMind, 2026b) judge scores the candidate and the target *independently* rather than side by side, which removes the position bias of pairwise comparison. Scoring uses a compact set of task- and content-type-specific aspects rated from 0 to 10. Text-to-image samples are rated on content faithfulness, readability, and a style-specific format aspect for academic, poster, or infographic figures, and the three reference-conditioned tasks replace the format aspect with an input-fidelity aspect tailored to how each task uses its conditioning input. A weighted mean turns the per-aspect scores into one total per image, and the candidate's margin over the target yields a verdict $o_i \in \{\textit{Model}, \textit{Tie}, \textit{Human}\}$ under a calibrated tie band. The bench-level score averages the $\{100, 50, 0\}$ mapping of these verdicts, and on academic text-to-image inputs it reduces to a PaperBanana-style referenced judge. A blind human study on a random sample confirms that this metric tracks human preference (Appendix H). Full prompts are in Appendix G.

5 EXPERIMENTS

We evaluate CRAFTER on two benchmarks: PaperBanana-Bench (Zhu et al., 2026a), which covers text-to-image generation of academic figures, and CRAFTBENCH, which extends coverage to three figure types and four input conditions. Both benchmarks are scored by referenced VLM-as-judge protocols that report a lenient win-rate against the human-drawn target (§4.3). We compare against standalone generators and agentic frameworks; to isolate the effect of orchestration design, all agentic methods share the same image-generation backbone (Nano Banana 2) and vision-language model (Gemini 3.1 Pro (Google DeepMind, 2026a)). Full configuration details are in Appendix A.

5.1 MAIN RESULTS

Table 2 presents the full comparison. CRAFTER achieves the highest overall score on both benchmarks regardless of its backbone, leading the strongest agentic baseline under controlled comparison by 16.61 points on PaperBanana-Bench and 22.22 points on CRAFTBENCH, and only CRAFTER improves over its standalone generator on every quality dimension and task of both benchmarks. PaperBanana also improves over its backbone overall, but its gain shrinks sharply on the broader benchmark, from 22.60 points on PaperBanana-Bench to 8.06 points on CRAFTBENCH, and it slips below its backbone on the sketch task. This is the generalization failure identified in Section 1: a pipeline optimized for one figure type and input condition transfers poorly to broader settings.

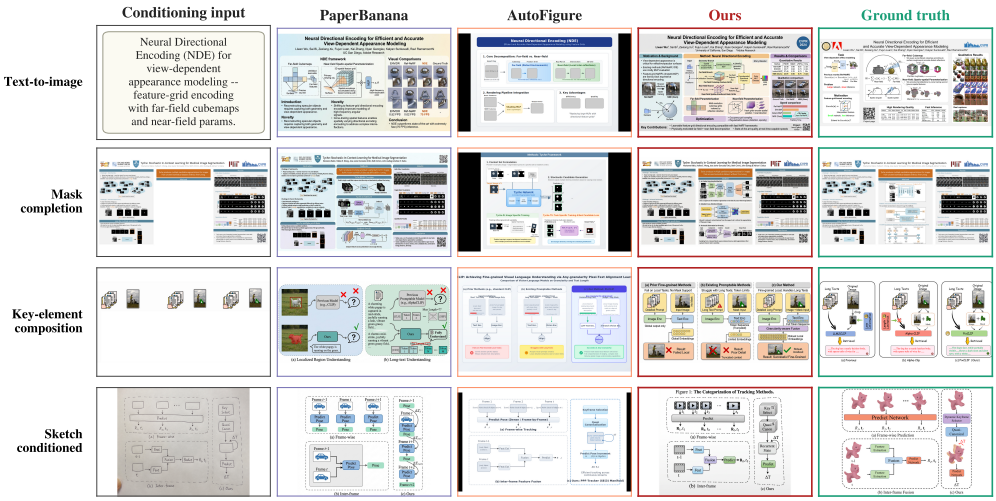


Figure 5: Qualitative comparison across different input conditions.

Table 3: Mechanism ablation on PaperBanana-Bench. Each row removes one mechanism from the full CRAFTER harness. **Bold**: best per column; Δ: overall gap vs. full CRAFTER.

Configuration	Faith.	Conc.	Read.	Aesth.	Overall	Δ
CRAFTER	**38.18**	**53.42**	**47.77**	**64.21**	**50.34**	
w/o plan exploration	28.42	51.20	38.30	60.10	41.78	−8.56
w/o corrective layer	27.18	49.66	36.86	58.45	41.44	−8.90
w/o refinement loop	30.07	51.97	41.55	61.80	44.86	−5.48
w/o directive critic	31.20	52.91	42.55	63.18	45.30	−5.04

AutoFigure falls below its backbone. On CRAFTBENCH, CRAFTER's lead over PaperBanana also holds when both are rescored by judges from two other model families (Appendix F.1).

Across its two backbones, CRAFTER attains the best score in every column of both benchmarks, including all four quality dimensions of PaperBanana-Bench and all four tasks of CRAFTBENCH (Figure 5). With either backbone, it also outscores every baseline within each figure style (Appendix B.5). No baseline surpasses CRAFTER in any column, and the strongest non-CRAFTER results are scattered across different systems and conditions, so each baseline's strength is confined to specific inputs. The harness leaves the generator's output ceiling unchanged but makes the pipeline more general and more reliable on diverse inputs. The two CRAFTER rows in Table 2 show that its overall score is largely independent of the executor: replacing Nano Banana 2 with Nano Banana Pro shifts the overall score by only 0.34 points on PaperBanana-Bench and 2.15 points on CRAFTBENCH, so stronger generators can be incorporated without modification. CRAFTER is not uniformly successful; Appendices K and L analyze representative success and failure cases.

5.2 ABLATION AND ANALYSIS

5.2.1 ABLATION STUDY

We conduct an ablation study on PaperBanana-Bench to test the contribution of each mechanism independently, by removing one at a time from the full CRAFTER pipeline. Results are shown in Table 3. Every removal degrades the overall score, with drops ranging from 5.04 to 8.90 points.

Restricting CRAFTER to a single candidate plan ($K=1$) incurs an 8.56 point drop (*w/o plan exploration*). Faithfulness and readability suffer the most among the four quality dimensions, because a wrong framing decision early on, such as rendering a comparison grid as a block diagram, propagates through every subsequent refinement round with no opportunity to escape. The *w/o corrective layer* experiment shows that replacing typed edits with free-text revision instructions costs 8.90 points

Table 4: Editable-output evaluation on 80 CRAFTER outputs. Scores range from 0 to 10 and are averaged over three VLM judges.

System	Pos.	Color	Text	Icon	Arrow	Style	Overall
Raster-to-editable methods							
Edit-Banana	4.21	4.93	2.86	4.97	4.18	4.32	3.69
AutoFigure-Edit	6.92	7.41	6.04	6.78	6.39	7.00	6.91
CRAFTEDITOR	**8.10**	**8.34**	**7.61**	**8.07**	**7.83**	**8.12**	**8.04**
Ablations							
w/o agentic cleaning	7.84	8.12	7.32	7.69	7.55	7.83	7.71
w/o iterative composition	6.05	6.41	5.32	5.94	5.71	6.10	5.89

overall. This result supports the concern raised in Section 1 that unstructured corrections accumulate contradictions across rounds. The *verify-then-refine loop* and its *directive critic* contribute 5.48 and 5.04 points, respectively. Removing the loop shows that iterative correction is needed to repair localized errors that survive the first generation. The directive critic's separate gain shows that per-dimension diagnostics provide actionable targets that scalar scores cannot: without them, the reviser still iterates but lacks a specific failure to address. We further analyze the scaling behavior of K and T and the computational cost in Appendices C.1 and C.2.

5.2.2 EDITABLE-OUTPUT QUALITY

Raster outputs cannot be locally revised, so we evaluate CRAFTEDITOR's ability to convert CRAFTER rasters into editable SVGs. We compare against Edit-Banana (BIT-DataLab Contributors, 2026) and AutoFigure-Edit (Lin et al., 2026) on seven axes, scored by an ensemble of three VLM judges over a held-out subset of 80 CRAFTER outputs. Setup details are in Appendices A.2 and D, and Appendix I complements these fidelity scores with a small human editing study.

Baseline comparison. CRAFTEDITOR leads on every evaluation axis (Table 4; Figure A7), achieving an overall score of 8.04 against 6.91 for AutoFigure-Edit and 3.69 for Edit-Banana. The margin over AutoFigure-Edit is widest on the structural axes (text and arrows) where precise coordinate reasoning and iterative correction matter most, as the examples in Figure A7 confirm. Two complementary designs account for this margin: the extraction phase supplies the composer with clean per-element assets rather than noisy crops, and the hybrid critic of the composition phase catches structural errors, such as misplaced arrow endpoints, across refinement rounds.

Design effectiveness. The ablation rows in Table 4 quantify each design's contribution. Removing iterative composition causes a sharp and uniform drop across all seven axes (overall -2.15), showing that critic-driven revision turns a brittle one-shot SVG into a faithful reproduction. Removing agentic cleaning yields a smaller but consistent effect (overall -0.33), with the largest per-axis drop falling on icons, where clean extraction of overlapping visual assets is most critical. Together, both ablations confirm that the two designs are jointly necessary for the CRAFTEDITOR.

6 CONCLUSION

We have presented a harness-based approach to scientific figure authoring addressing two practical gaps left by existing systems: limited generalization across figure types and input conditions, and the inability to produce editable outputs. CRAFTER and CRAFTEDITOR instantiate this harness for figure generation and raster-to-SVG conversion respectively, and CRAFTBENCH provides the first benchmark for cross-type, cross-condition evaluation. CRAFTER outperforms all baselines on both PaperBanana-Bench and CRAFTBENCH, and CRAFTEDITOR leads prior methods on every axis. Because the harness is executor-agnostic, it admits stronger future generators without modification, and we expect the pattern to extend to structured-output domains beyond scientific figures.

AI USE STATEMENT

Generative AI models are part of our evaluation: vision-language models serve as the judges that score both benchmarks and the CRAFTEDITOR outputs (Section 4.3 and Appendix E), and they also assisted with the filtering and input construction of CRAFTBENCH under full human review (Appendix B). In preparing the paper, we used generative AI tools to polish the writing, such as correcting grammar and improving wording, and to check the formatting. The design of the experiments and the interpretation of the results were carried out by the authors. The authors reviewed all AI-assisted content and take full responsibility for the paper.

ETHICS STATEMENT

CRAFTER and CRAFTEDITOR can produce figures that look polished yet contain incorrect labels, invented or missing components, or a structure that misrepresents the method or result they depict. Their outputs are drafts for authors to verify, not finished scientific artifacts, and responsibility for a published figure remains with the authors who publish it. Our evaluation penalizes this failure. The CRAFTBENCH judge weights content faithfulness at 3.0, against 1.0 to 1.5 for readability and format (Appendix F), and is instructed to score plausible-looking but wrong content low (Figure A3). These checks reduce the risk without removing it, and Appendix L documents outputs that drop panels named in the caption or fill a masked region with content that does not continue the preserved figure. CRAFTBENCH is built entirely from publicly available sources, namely arXiv preprints, award-tier conference posters, and a public research blog. We use this material for non-commercial research evaluation, retain only figures with their captions and surrounding text, and release sample identifiers and provenance rather than redistributing source documents. The human evaluation of Appendix H was carried out by three graduate-level annotators who were informed of the task and compensated at a local hourly rate above the regional minimum wage, and two further annotators carried out the editing study of Appendix I.

REFERENCES

Anthropic. Claude Opus 4.6. Anthropic, 2026a. https://www.anthropic.com/news/claude-opus-4-6.

Anthropic. Claude Opus 4.7. Anthropic, 2026b. https://www.anthropic.com/news/claude-opus-4-7.

Anthropic. Claude Sonnet 4.6. Anthropic, 2026c. https://www.anthropic.com/news/claude-sonnet-4-6.

Jonas Belouadi, Anne Lauscher, and Steffen Eger. AutomaTikZ: Text-guided synthesis of scientific vector graphics with TikZ. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

BIT-DataLab Contributors. Edit-Banana: Make the uneditable, editable. https://github.com/BIT-DataLab/Edit-Banana, 2026.

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

Nghi D. Q. Bui. Building effective AI coding agents for the terminal: Scaffolding, harness, context engineering, and lessons learned. *arXiv preprint arXiv:2603.05344*, 2026.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. SAM 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

Yifan Chang, Yukang Feng, Jianwen Sun, Jiaxin Ai, Chuanhao Li, S. Kevin Zhou, and Kaipeng Zhang. SridBench: Benchmark of scientific research illustration drawing of image generation model, 2025. URL https://arxiv.org/abs/2505.22126.

Google DeepMind. Introducing Nano Banana Pro. Google Blog, 2025. https://blog.google/innovation-and-ai/products/nano-banana-pro/.

Google DeepMind. Gemini 3.1 Pro. Google DeepMind, 2026a. https://deepmind.google/models/gemini/pro/.

Google DeepMind. Gemini 3.5 Flash. Google DeepMind, 2026b. https://deepmind.google/models/gemini/flash/.

Google DeepMind. Nano Banana 2: Combining Pro capabilities with lightning-fast speed. Google Blog, 2026c. https://blog.google/innovation-and-ai/technology/ai/nano-banana-2/.

Christian Greisinger and Steffen Eger. TikZilla: Scaling text-to-TikZ with high-quality data and reinforcement learning. *arXiv preprint arXiv:2603.03072*, 2026.

Ziyi Guo, Zhou Liu, and Wentao Zhang. Paper2SysArch: Structure-constrained system architecture generation from scientific papers. *arXiv preprint arXiv:2511.18036*, 2025.

Dikshant Kukreja, Kshitij Sah, Karan Goyal, Mukesh Mohania, and Vikram Goyal. CAGE: Bridging the accuracy-aesthetics gap in educational diagrams via code-anchored generative enhancement. *arXiv preprint arXiv:2604.09691*, 2026.

Zhen Lin, Qiujie Xie, Minjun Zhu, Shichen Li, Qiyao Sun, Enhao Gu, Yiran Ding, Ke Sun, Fang Guo, Panzhong Lu, Zhiyuan Ning, Yixuan Weng, and Yue Zhang. AutoFigure-Edit: Generating editable scientific illustration. *arXiv preprint arXiv:2603.06674*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

OpenAI. Introducing GPT-5.4. OpenAI, 2026a. https://openai.com/index/introducing-gpt-5-4/.

OpenAI. Introducing GPT-5.5. OpenAI, 2026b. https://openai.com/index/introducing-gpt-5-5/.

OpenAI. Introducing ChatGPT Images 2.0. OpenAI Blog, 2026c. https://openai.com/index/introducing-chatgpt-images-2-0/.

Linyue Pan, Lexiao Zou, Shuo Guo, Jingchen Ni, and Hai-Tao Zheng. Natural-language agent harnesses. *arXiv preprint arXiv:2603.25723*, 2026.

Shuzheng Si, Wentao Ma, Haoyu Gao, Yuchuan Wu, Ting-En Lin, Yinpei Dai, Hangyu Li, Rui Yan, Fei Huang, and Yongbin Li. SpokenWOZ: A large-scale speech-text benchmark for spoken task-oriented dialogue agents. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=viktK3nO5b.

Shuzheng Si, Haozhe Zhao, Gang Chen, Yunshui Li, Kangyang Luo, Chuancheng Lv, Kaikai An, Fanchao Qi, Baobao Chang, and Maosong Sun. GATEAU: Selecting influential samples for long context alignment. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 7380–7411, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.375. URL https://aclanthology.org/2025.emnlp-main.375/.

Shuzheng Si, Haozhe Zhao, Yu Lei, Qingyi Wang, Dingwei Chen, Zhitong Wang, Zhenhailong Wang, Kangyang Luo, Zheng Wang, Gang Chen, Fanchao Qi, Minjia Zhang, and Maosong Sun. From context to skills: Can language models learn from context skillfully?, 2026a. URL https://arxiv.org/abs/2604.27660.

Shuzheng Si, Haozhe Zhao, Kangyang Luo, Gang Chen, Fanchao Qi, Minjia Zhang, Baobao Chang, and Maosong Sun. A goal without a plan is just a wish: Efficient and effective global planner training for long-horizon agent tasks, 2026b. URL https://arxiv.org/abs/2510.05608.

Tao Sun, Enhao Pan, Zhengkai Yang, Kaixin Sui, Jiajun Shi, Xianfu Cheng, Tongliang Li, Wenhao Huang, Ge Zhang, Jian Yang, and Zhoujun Li. P2P: Automated paper-to-poster generation and fine-grained benchmark. *arXiv preprint arXiv:2505.17104*, 2025.

Yinghao Tang, Xueding Liu, Boyuan Zhang, Tingfeng Lan, Yupeng Xie, Jiale Lao, Yiyao Wang, Haoxuan Li, Tingting Gao, Bo Pan, Luoxuan Weng, Xiuqi Huang, Minfeng Zhu, Yingchaojie Feng, Yuyu Luo, and Wei Chen. IGenBench: Benchmarking the reliability of text-to-infographic generation. *arXiv preprint arXiv:2601.04498*, 2026.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

Haoyue Yang, Xuanle Zhao, Xuexin Liu, Feibang Jiang, and Yao Zhu. OmniDiagram: Advancing unified diagram code generation via visual interrogation reward. In *Findings of the Association for Computational Linguistics (ACL Findings)*, 2026. arXiv:2604.05514.

Justin Young. Effective harnesses for long-running agents. Anthropic Blog, 2025. https://www.anthropic.com/engineering/effective-harnesses-for-long-running-agents.

Z.ai. GLM-Image: Auto-regressive for dense-knowledge and high-fidelity image generation. https://github.com/zai-org/GLM-Image, 2026.

Abhay Zala, Han Lin, Jaemin Cho, and Mohit Bansal. DiagrammerGPT: Generating open-domain, open-platform diagrams via LLM planning. In *Conference on Language Modeling (COLM)*, 2024.

Tong Zhang, Honglin Lin, Zhou Liu, Chong Chen, and Wentao Zhang. SciFlow-Bench: Evaluating structure-aware scientific diagram generation via inverse parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. arXiv:2602.09809.

Hao Zheng, Xinyan Guan, Hao Kong, Jia Zheng, Weixiang Zhou, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. PPTAgent: Generating and evaluating presentations beyond text-to-slides, 2025. URL https://arxiv.org/abs/2501.03936.

Dawei Zhu, Rui Meng, Yale Song, Xiyu Wei, Sujian Li, Tomas Pfister, and Jinsung Yoon. PaperBanana: Automating academic illustration for AI scientists. *arXiv preprint arXiv:2601.23265*, 2026a.

Minjun Zhu, Zhen Lin, Yixuan Weng, Panzhong Lu, Qiujie Xie, Yifan Wei, Sifan Liu, Qiyao Sun, and Yue Zhang. AutoFigure: Generating and refining publication-ready scientific illustrations. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026b. arXiv:2602.03828.

APPENDIX

This appendix is organized as follows.

- In Appendix A, we give the experimental setup: CRAFTER on PaperBanana-Bench and CRAFTBENCH (Appendix A.1) and CRAFTEDITOR on raster-to-vector conversion (Appendix A.2).
- In Appendix B, we detail CRAFTBENCH: source pools (Appendix B.1), the quality gates (Appendix B.2), the reference-conditioned task construction (Appendix B.3), the sensitivity to the mask-completion task (Appendix B.4), and scores by figure style (Appendix B.5).

- In Appendix C, we provide CRAFTER implementation details, including the scaling behavior of K and T (Appendix C.1), the computational cost (Appendix C.2), and an agent backbone robustness study (Appendix C.3).
- In Appendix D, we present CRAFTEDITOR implementation details and ablations.
- In Appendix E, we describe the CRAFTEDITOR judge-ensemble protocol.
- In Appendix F, we give the full CRAFTBENCH evaluation protocol and rescore CRAFTER and PaperBanana with judges from other model families (Appendix F.1).
- In Appendix G, we reproduce the full judge prompts.
- In Appendix H, we report a human evaluation validating the automatic judge.
- In Appendix I, we report a human editing study of the editable outputs.
- In Appendix J, we discuss limitations.
- In Appendix K, we present additional qualitative case studies.
- In Appendix L, we analyze representative failure cases of CRAFTER.

A EXPERIMENTAL SETUP

This appendix details the experimental setup summarized in Section 5.

A.1 CRAFTER ON PAPERBANANA-BENCH AND CRAFTBENCH

Benchmarks and metric. We evaluate on PaperBanana-Bench (Zhu et al., 2026a), which covers text-to-image generation of 292 academic methodology figures, and on our proposed CRAFTBENCH (n=279; Section 4). PaperBanana-Bench is scored under its official protocol with Gemini 3.1 Pro (`gemini-3.1-pro-preview`) as the VLM judge. CRAFTBENCH is scored under the per-image referenced protocol of Section 4.3 and Appendix F with Gemini 3.5 Flash. Both report the lenient win-rate, the average of the per-sample $\{100, 50, 0\}$ verdict mapping.

Baselines. We compare against three groups of methods: two open-source generators (GLM-Image (Z.ai, 2026), Qwen-Image (Wu et al., 2025)), three closed-source generators (GPT-Image-2 (OpenAI, 2026c), Nano Banana 2 (Google DeepMind, 2026c), Nano Banana Pro (Google DeepMind, 2025)), and two agentic frameworks (PaperBanana (Zhu et al., 2026a) and AutoFigure (Zhu et al., 2026b)). Every method, standalone or agentic, receives the same inputs: the caption, the source paper-text, and, for a reference-conditioned sample, its conditioning input (the masked figure and its region label for mask-completion, the element skeleton for key-element composition, or the sketch for sketch-conditioned generation). This input is passed to each image-generation model through its multimodal interface, the same route CRAFTER's executor uses. Standalone generators are queried once with default decoding parameters and no retries.

Controlled comparison. To isolate the effect of orchestration design, all agentic methods share the same image-generation backbone (Nano Banana 2) and the same vision-language model (`gemini-3.1-pro-preview`) for all vision-dependent agents. We additionally report CRAFTER and PaperBanana with Nano Banana Pro, the former to verify executor pluggability (Section 3.1). Full model assignments and per-agent configurations are in Appendix C.

Inference and judging. Each benchmark is run end-to-end once per method. PaperBanana-Bench uses the official per-dimension judging configuration of Zhu et al. (2026a). On CRAFTBENCH, each image is scored once per applicable aspect using the prompts of Appendix G, with the judge queried at `temperature = 0` under a fixed seed and up to 8 retries with exponential backoff. A missing or corrupt generation counts as an automatic *Human* verdict and is never dropped. Random seeds for the harness loop are fixed across the agentic methods.

A.2 CRAFTEDITOR ON RASTER-TO-VECTOR CONVERSION

Subset. CRAFTEDITOR is evaluated on a random subset of 80 rasters drawn from CRAFTER's outputs across both PaperBanana-Bench and CRAFTBENCH, balanced across academic, poster, and

infographic figure types so that each type contributes a comparable share. The subset is held out of the main CRAFTER comparison and is used only for the editable-output evaluation.

Baselines. We compare against two raster-to-vector systems that make automated figure outputs editable: Edit-Banana (BIT-DataLab Contributors, 2026), which converts the raster through a SAM 3 (Carion et al., 2025) segmentation pipeline and writes DrawIO cells, and AutoFigure-Edit (Lin et al., 2026), which detects icons with SAM 3 and emits a full SVG in a single LLM call. Both receive the same input raster as CRAFTEDITOR and use their public default settings.

Judge ensemble. A three-VLM ensemble scores each output on seven axes (*position*, *color*, *text*, *icon*, *arrow*, *style*, plus a holistic *overall*) on a scale from 0 to 10. The three judges are Gemini 3.1 Flash-Lite, GPT-5.4, and Doubao-Seed-2.0-Pro, queried independently with the same prompt and aggregated by mean. A single-judge retry rule re-queries any model returning an overall score below 3.0 once; the retry replaces the original when its overall is higher. Full details are in Appendix E.

Ablations. The two CRAFTEDITOR ablations (no agentic cleaning; no iterative composition) use the same 80-sample subset and three-VLM ensemble; per-stage parameter changes are in Appendix D.

B CRAFTBENCH: DATASET DETAILS

B.1 SOURCE POOL COMPOSITION

The 279 samples in CRAFTBENCH are drawn from five source pools (Table A1). The academic pool is split across two arXiv crawls: a broad-domain crawl targeted at general method-figure coverage, and a method/architecture crawl targeted at the figure types that underpin the key-element-composition and sketch-conditioned-generation constructions (figures whose caption contains *overview*, *pipeline*, *architecture*, *method*, *approach*, or *framework*).

Table A1: Source pools feeding CRAFTBENCH. Each sample passes the seven-stage quality pipeline of Appendix B.2; NeurIPS spotlight and oral posters were also crawled, but none survived.

Pool	**n**	**Provenance**
arXiv (broad-domain crawl)	84	cs.LG / cs.CV / cs.CL / cs.RO / cs.AI / cs.NE / cs.HC / cs.IR / cs.GR / cs.SD / cs.DB / cs.MM / cs.CR / eess.IV / eess.AS / stat.ML / q-bio (BM,NC,QM) / physics (bio-ph, med-ph)
arXiv (method/architecture crawl)	56	method-figure crawl restricted to overview / pipeline / architecture / method / approach / framework captions
CVPR posters (highlight + award)	55	CVPR highlight and award tier
ICLR posters (oral + spotlight)	54	ICLR oral and spotlight via OpenReview
Lil'Log (Lilian Weng's blog)	30	deep-dive ML topics
Total	**279**	

B.2 QUALITY GATES

Every candidate passes a seven-stage pipeline before entering CRAFTBENCH:

1. **Caption keyword filter** (G1): requires figure-type language and method-related keywords (e.g., *overview*, *architecture*, *pipeline*) in the caption.
2. **Strict content classifier** (G2): a vision-language classifier assigns each figure to one of 15 fine-grained types; only *diagram*, *illustration showing method*, *architecture*, and *teaser* are accepted. Photographs, statistical charts, screenshots, equation-only renders, and tables are rejected.
3. **Complexity rescore** (G3): a vision-language rubric verifies that the figure is worth recreating as a drawing, exhibits sufficient design richness (score $\geq 4/5$), contains at least 8 distinct named components, and would take an estimated 10+ minutes to recreate manually.
4. **Alignment verification** (G4): a vision-language check verifies that at least 70% of authored visual claims match the figure content, at least 60% of proposed edit targets are feasible, and the caption alignment score reaches $\geq 3/5$.

5. **First-pass quality assurance** (G5): a vision-language reviewer flags cropping artifacts, watermarks, low resolution, caption mismatch, and edit-target referent issues.
6. **Evidence-required quality assurance** (G6): a stricter second pass in which every flag must cite direct pixel-region evidence with confidence $\geq 4/5$; unsupported flags are auto-discarded.
7. **Manual review** (G7): human inspection of every flagged sample plus all reference-conditioned task inputs. The interface and acceptance rule are described in Appendix B.3.

B.3 Reference-Conditioned Task Construction

For each of the three reference-conditioned tasks, conditioning inputs are constructed from the source figures through a semi-automatic pipeline followed by manual quality assurance. Every input is inspected and either hand-edited or hand-confirmed before it enters the benchmark, and the counts below reflect samples that survived the full quality pipeline including unanimous human agreement.

Mask-completion (n=30). A vision-language model proposes a semantic region whose removal still leaves the figure interpretable, and the region is masked to pure white so that the unmasked pixels stay identical to the ground truth. The region's identity is recorded as a short natural-language label whose wording is hand-polished for every sample. Mask area ranges from roughly 20% to over 90% of the figure, averaging about 40%. The masks span three layouts (single rectangle 12, multiple rectangles 10, single unlabeled region 8), with 27 inputs hand-edited by re-cropping the bounding box or repainting the region and 3 hand-confirmed without modification. Preservation of the unmasked pixels can be checked directly against the input, and the input-fidelity aspect reserves scores of 8 to 10 for outputs that fill the masked region correctly and leave the rest unchanged (Figure A3).

Key-element composition (n=30). An image-editing model extracts only the icon-level placeholders that define the figure's spatial logic into a minimal, representative element skeleton, with text labels and connecting arrows stripped. Small random displacement is applied to each extracted element to discourage trivial gap-filling. A human annotator reviews each result, deleting incorrectly extracted elements and equalizing difficulty (29 hand-edited, 1 hand-confirmed).

Sketch-conditioned generation (n=40). A layout reference is prepared in one of three sketch families: hand-drawn or pen sketches (n=15), AI-drafted rough sketches conditioned on the caption and surrounding context, deliberately not on source pixels, so that the sketch stays rough rather than retracing the source (n=14), and SVG wireframes generated from the source figure's vector code and rasterized (n=11). Every sketch input passes human quality assurance.

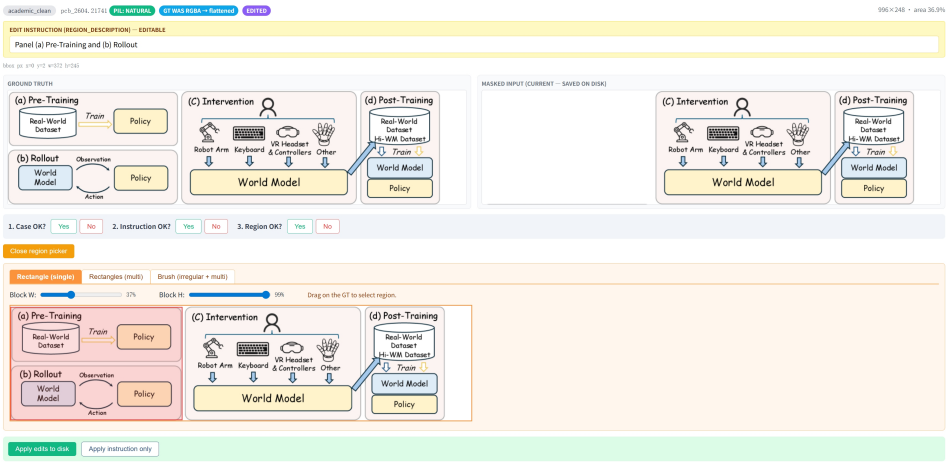
Manual quality assurance. Every reference-conditioned task input is reviewed through a dedicated browser annotation interface, one per task family, with three graduate-level annotators independently inspecting each sample. A sample is accepted into CraftBench only when all three annotators agree it is acceptable; any disagreement triggers a revision (hand-recrop, hand-paint, regenerate, or family swap) followed by re-review until unanimous agreement is reached. The three interfaces and their per-task operations are summarized in Figure A1: mask-completion uses a drag-to-mask overlay with an editable region label; key-element composition uses rectangle, multi-rectangle, and brush erasers to remove leftover text and arrows after automatic extraction; sketch-conditioned generation uses the same erasers plus a one-click family-switch that redraws the layout reference under a different sketch family when the current one is rejected.

B.4 Sensitivity to the Mask-Completion Task

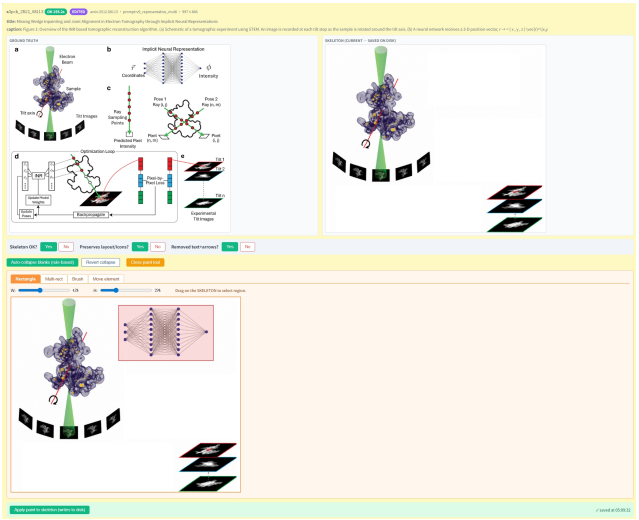
Mask-completion contributes 30 of the 279 CraftBench samples. Table A2 recomputes the overall score of Crafter and PaperBanana, the strongest agentic baseline, over the remaining 249 samples. Because the overall score averages verdicts over samples, these values follow directly from the per-task columns of Table 2 and the task sizes in Section 4.2. The gap widens under both backbones, and with Nano Banana 2 Crafter still leads every remaining task, with 48.32 vs. 18.72 on text-to-image, 70.00 vs. 60.00 on sketch-conditioned generation, and 40.00 vs. 31.67 on key-element composition.

B.5 Scores by Figure Style

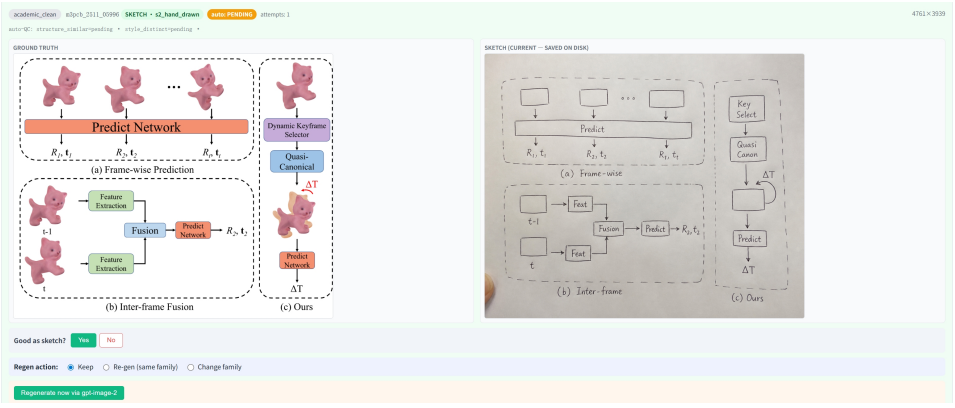
Table A3 splits the CraftBench scores of Table 2 by figure style, using the same outputs and judge verdicts; weighting the three styles by sample count recovers the overall score of Table 2. Academic



(a) Mask-completion



(b) Key-element composition



(c) Sketch-conditioned generation

Figure A1: Manual annotation interfaces for the three reference-conditioned task constructions. Each interface combines the auto-generated conditioning input with task-specific drawing tools (drag-to-mask overlay, rectangle/brush erasers, family-switch regeneration); three graduate-level annotators inspect every sample and a unanimous-agreement rule decides acceptance.

Table A2: Overall CRAFTBENCH score (%) with and without the 30 mask-completion samples. Gaps are computed before rounding.

Method	**All tasks** (n=279)	**w/o mask-completion** (n=249)
PaperBanana (w/ Nano Banana 2)	27.96	26.91
CRAFTER (w/ Nano Banana 2)	50.18	50.80
Gap	22.22	23.90
PaperBanana (w/ Nano Banana Pro)	29.03	29.72
CRAFTER (w/ Nano Banana Pro)	52.33	53.61
Gap	23.30	23.90

figures hold all 40 sketch-conditioned and 30 key-element samples besides 59 text-to-image and 11 mask-completion samples, while posters (97 and 12) and infographics (23 and 7) contain only the latter two tasks, so differences across styles partly reflect task mix.

CRAFTER with either backbone outscores every baseline in every style. Under the controlled comparison with Nano Banana 2, CRAFTER leads PaperBanana by 18.57 points on academic figures, 27.52 on posters, and 20.00 on infographics. On posters, no baseline exceeds 10 points, while CRAFTER reaches 33.94 and 36.70. The largest gain over the standalone generator is on infographics (+46.67 and +45.00), but with only 30 samples this style has the widest uncertainty.

Table A3: CRAFTBENCH scores (%) by figure style. **Bold** marks column-best; Δ is the gap between CRAFTER and its standalone generator, computed before rounding. *GPT-Image-2 returned valid outputs for 260 of 279 inputs.

Method	**Academic** (n=140)	**Poster** (n=109)	**Infographic** (n=30)	**Overall** (n=279)
Standalone generators (open-source)				
GLM-Image	0.00	0.00	0.00	0.00
Qwen-Image	0.71	0.00	0.00	0.36
Standalone generators (closed-source)				
GPT-Image-2*	21.43	7.80	18.33	15.77
Nano Banana 2	31.79	1.83	30.00	19.89
Nano Banana Pro	33.57	3.67	38.33	22.40
Agentic frameworks				
AutoFigure (w/ Nano Banana 2)	2.86	0.00	6.67	2.15
PaperBanana (w/ Nano Banana 2)	38.57	6.42	56.67	27.96
PaperBanana (w/ Nano Banana Pro)	39.29	9.63	51.67	29.03
CRAFTER (w/ Nano Banana 2)	57.14	33.94	76.67	50.18
Δ vs. Nano Banana 2	+25.36	+32.11	+46.67	+30.29
CRAFTER (w/ Nano Banana Pro)	**57.86**	**36.70**	**83.33**	**52.33**
Δ vs. Nano Banana Pro	+24.29	+33.03	+45.00	+29.93

C CRAFTER: IMPLEMENTATION DETAILS

Model configuration. For fair comparison, all agentic methods share the same image-generation backbone and vision-language model. The default backend $\mathcal{E}$ is Nano Banana 2 (`gemini-3.1-flash-image-preview`); we additionally report CRAFTER with Nano Banana Pro (`gemini-3.0-pro-image-preview`). Vision-dependent agents (critic $\mathcal{V}$, convergence judge) use `gemini-3.1-pro-preview` across CRAFTER, PaperBanana, and AutoFigure alike. For the language-only agents (intent reasoner, plan generator $\mathcal{D}$, specification refiner $\mathcal{R}$), CRAFTER adopts `claude-opus-4-6` (Anthropic, 2026a). For the open-source baselines, GLM-Image is evaluated directly; Qwen-Image uses Qwen-Image-2512 for text-to-image and Qwen-Image-Edit-2511 for reference-conditioned tasks.

Pipeline orchestration. The five agents coordinate around the evolving figure specification $\mathcal{S}$ (Section 3.1). $\mathcal{E}$ supplies raster outputs but is not itself an agent. All agents read and write $\mathcal{S}$;

free-text addenda from one agent never reach another agent's prompt directly. $\mathcal{D}$ emits $K \in \{1, 2, 3\}$ visual-style keys from an 11-key vocabulary (banner, multi-column grid, numbered-step sequence, etc.); the convergence judge bounds the loop at $T{=}3$ rounds; the early-exit gate fires when content accuracy and role conformity both exceed $6.5/10$ and no pixel artifact is detected.

Image-generation routing. For text-to-image samples, $\mathcal{E}$ is called once per plan (Si et al., 2023; 2026b) with a single text prompt. For reference-conditioned samples, $\mathcal{E}$ is called via the multimodal interface with the reference image as the first item and a task-specific instruction as the text. Adding a new reference-conditioned task requires one additional branch in the instruction builder plus one entry in the reference-image role-hint table; no other pipeline code changes.

Typed corrective edits. $\mathcal{R}$ emits typed edits to $\mathcal{S}$ from a fixed vocabulary of structured operations (adding layout constraints, banning artifact categories, resizing or demoting named elements). Each edit composes through $\mathcal{S}$ rather than as a free-text prompt addendum, so the next round's prompt builder sees a coherent typed delta. $\mathcal{V}$ emits a directive diagnostic d_t containing per-dimension scores along six quality axes (content accuracy, layout coherence, text legibility, role conformity, aesthetic quality, artifact severity), an issues list, a suggestions list, and a revised figure description; $\mathcal{R}$ reads only the issues and suggestions to decide which edits to apply.

Convergence judge. The convergence judge applies hard rules (iteration cap, content-score threshold, overall-score threshold, pixel-artifact detection) when applicable, and falls back to a vision-language acceptance call otherwise. After the loop terminates, the judge selects the highest-scoring artifact a^* across all rounds, with a post-correction pass (OCR-based typo repair on rendered text, plus a quality-guard revert when post-correction introduces an artifact).

C.1 Scaling Behavior of K and T

We vary the number of candidate plans K and refinement rounds T independently on PaperBanana-Bench to understand each dimension's contribution. Results are shown in Table A4.

Table A4: Effect of plan candidates K and refinement rounds T on PaperBanana-Bench. Each group varies one factor while holding the other fixed. Δ: gap vs. full configuration (last row).

	Faith.	Conc.	Read.	Aesth.	Overall	Δ
Varying K (fixed $T{=}3$)						
$K{=}1$	28.42	51.20	38.30	60.10	41.78	-8.56
$K{=}3$	34.25	56.16	48.45	66.44	48.97	-1.37
Varying T (fixed $K{=}$adaptive)						
$T{=}1$	30.07	51.97	41.55	61.80	44.86	-5.48
Full ($K{=}$adaptive, $T{=}3$)	38.18	53.42	47.77	64.21	50.34	

Increasing K from 1 to 3 yields the largest single gain ($+7.19$ points), so plan-level diversity matters most for escaping structurally unsuitable framings. Moving from fixed $K{=}3$ to adaptive K adds a further $+1.37$ points; the adaptive strategy allocates more candidates to complex, multi-constraint inputs and fewer to simple ones, producing a $+3.93$ point faithfulness gain over $K{=}3$. Increasing T from 1 to 3 yields $+5.48$ points. K governs whether the pipeline starts from a viable framing, and T governs whether localized errors in that framing get corrected.

C.2 Computational Cost

Table A5 reports the per-figure inference cost for CRAFTER, PaperBanana, and CRAFTEDITOR.

CRAFTER costs two to three times more per figure than PaperBanana, trading inference budget for multi-variant plan exploration and iterative refinement. This overhead is modest in context: a publication-quality figure typically requires hours of manual effort, and the total cost of generating all 279 CRAFTBENCH samples is under $90. CRAFTEDITOR adds $0.85 per raster-to-SVG conversion, with the majority of cost attributable to LLM output tokens during iterative SVG refinement.

Table A5: Per-figure generation cost.

System	Cost
AutoFigure (w/ Nano Banana 2)	$0.06
PaperBanana (w/ Nano Banana 2)	$0.11
CRAFTER (w/ Nano Banana 2)	$0.25
CRAFTER (w/ Nano Banana Pro)	$0.32
CRAFTEDITOR (per conversion)	$0.85

C.3 AGENT BACKBONE ROBUSTNESS

CRAFTER uses `claude-opus-4-6` (Anthropic, 2026a) for its language-only agents (the planner) and `gemini-3.1-pro-preview` for its vision-dependent agents (the VLM). Table A6 swaps these backbones on CRAFTBENCH while keeping Nano Banana 2 as the image-generation backend, with `claude-opus-4-7` (Anthropic, 2026b), `gemini-3.1-pro-preview`, Gemini 3.5 Flash (Google DeepMind, 2026b), and `gpt-5.5` (OpenAI, 2026b) as alternatives. The planner affects the score: running every agent on Gemini 3.1 Pro, the VLM shared by all agentic methods, lowers it from 50.2 to 38.4. Every configuration, including this one, still scores above the strongest baseline on CRAFTBENCH (29.0, PaperBanana with Nano Banana Pro; Table 2), so CRAFTER leads without the Claude planner, although by a smaller margin. Moving the planner from Claude Opus 4.6 to 4.7 shifts the score by 2.2 points, so the result is not tied to a single model release.

Table A6: Agent backbone swaps on CRAFTBENCH (overall %), with Nano Banana 2 as the image-generation backend. Top: planner only; bottom: planner and VLM. The first row is the default.

Planner	VLM	Overall
Claude Opus 4.6	Gemini 3.1 Pro	**50.2**
Claude Opus 4.7	Gemini 3.1 Pro	48.0
Gemini 3.1 Pro	Gemini 3.1 Pro	38.4
Gemini 3.5 Flash	Gemini 3.5 Flash	40.7
GPT-5.5	GPT-5.5	39.1

D CRAFTEDITOR: IMPLEMENTATION DETAILS AND ABLATIONS

Extraction phase. A vision-language designer agent $\mathcal{D}$ inspects the input raster a^* and authors a per-figure keep/delete plan; an instructable image-editing executor $\mathcal{E}$ carries out the plan at the pixel level. A verify-then-refine wrapper runs at most $T{=}3$ iterations: a verifier $\mathcal{V}$ (a lightweight VLM, decoupled from the editor) inspects each cleaned candidate and either accepts it or returns a directive diagnostic (e.g., "the bottom-row icons were over-deleted; restore them; remove the page number instead") that triggers another round. Iteration-count distribution on dense posters: 47% converge at round 1, 46% at round 2, 7% at round 3.

Composition phase. After the processing phase (captioning, referring grounding, and per-element vector/raster classification), a language model generates two candidate SVG skeletons at decoding temperatures 0.20 and 0.45; the convergence judge picks the better candidate. Extracted assets are spliced into the placeholders, and $T{=}4$ refinement rounds run with the hybrid critic reporting per-axis scores (text presence, arrow endpoints, layout consistency, color drift). A best-so-far reversion guards against non-monotonic regressions: without it, approximately 30% of refinement iterations score lower than the immediately preceding iteration.

Provider abstraction. Four external services (LLM, image editor, segmentation, background removal) are wrapped behind interface adapters. Swapping a backend (e.g., the segmentation model for ablation) is a single configuration change.

Ablation setup. Two ablations target the two harness-instantiation phases of CRAFTEDITOR (Table 4), evaluated on the same 80-sample subset with the same three-VLM judge ensemble (Appendix E). *w/o agentic cleaning*: the extraction phase (Stage 1) is skipped; captioning runs on the original raster

and per-element extraction falls back to a segmentation-plus-background-removal alternative. *w/o iterative composition*: the composition-phase refinement loop is disabled (T=0); the skeleton with injected assets is returned directly without critic-driven revision.

Per-category extraction-phase analysis. CRAFTEDITOR wins or ties the "w/o agentic cleaning" ablation in 11 of 12 source categories under the headline judge ensemble. The single exception is a 3-sample text-to-image infographic subset, too small to support a conclusion.

E CRAFTEDITOR: JUDGE ENSEMBLE PROTOCOL

Models and prompt. The judge ensemble comprises three independent VLMs: Gemini 3.1 Flash-Lite, GPT-5.4, and Doubao-Seed-2.0-Pro. Each judge receives the original input raster and the rendered SVG as two image attachments and scores seven axes (*position*, *color*, *text*, *icon*, *arrow*, *style*, *overall*) on a scale from 0 to 10, returning a JSON response with per-axis scores and a structured issues list. All calls use temperature 0.15 and a 4,000-token output cap.

Aggregation. The headline score per sample is the mean of the three judges' *overall* scores. Any judge returning an *overall* score below 3.0 is re-queried once; if the retry scores higher, it replaces the original. The retry guards against an occasional outlier score from a single judge call.

Relationship to the generation rubrics. The seven-axis editable-output rubric is distinct from the generation rubrics used on PaperBanana-Bench and CRAFTBENCH (Section 4.3). Those rubrics score generation quality against a human-drawn target, whereas the seven-axis rubric scores reproduction fidelity of a raster-to-SVG conversion against the input raster. The two are reported in separate tables (Table 2 vs. Table 4) and should not be compared directly.

F EVALUATION PROTOCOL DETAILS

We score CRAFTBENCH with a per-image referenced protocol. For each sample, a Gemini 3.5 Flash judge (temperature 0, fixed seed) scores the generated figure and the human-drawn ground truth independently, one image at a time, against the same inputs the generator received (caption, paper context, and, for the reference-conditioned tasks, the input image). Scoring each image on its own rather than as an A/B pair avoids position bias. Each image is rated from 0 to 10 on a small aspect set chosen by task and content type: content faithfulness and readability on every sample, plus a style format aspect for text-to-image (academic, poster, or infographic) or an input-fidelity aspect for the three reference-conditioned tasks. The prompts, reproduced in Appendix G, are designed to penalize figures that look polished but are wrong. The shared scoring anchors (Figure A2) require the judge to list observed weaknesses before scoring, to re-examine the figure for a flaw before awarding 9 or 10, and to reward correctness and communication rather than surface polish. The definition of content faithfulness (Figure A3) directs the judge to check that every label is genuine and that no component is fabricated or missing, and to score plausible-looking but wrong content low.

A weighted mean of the aspect scores gives each image a single total, weighting content faithfulness and input fidelity most heavily (3.0) and readability and format least (1.0 to 1.5). The generated figure is judged *Model* when its total exceeds the ground truth's by more than a tie band of 0.30, *Human* when it trails by more than 0.30, and *Tie* otherwise. A missing generation counts as *Human*. The bench-level score is the lenient win-rate, the mean over samples of the $\{100, 50, 0\}$ mapping of these verdicts, and on academic text-to-image inputs it reduces to a PaperBanana-style referenced judge. PaperBanana-Bench is scored separately under its official protocol with Gemini 3.1 Pro.

F.1 JUDGES FROM OTHER MODEL FAMILIES

The default CRAFTBENCH judge, Gemini 3.5 Flash, belongs to the same model family as the Nano Banana backbones, so a judge preference for images from its own family could raise the scores of systems built on them. Within the controlled comparison, such a preference would not by itself favor one agentic system over another, since all of them render with Nano Banana 2 and use Gemini 3.1 Pro for their vision-dependent agents (Appendix A.1). We also test for it directly by rescoring the CRAFTBENCH outputs of CRAFTER and PaperBanana, both with Nano Banana 2, with GPT-5.4 (OpenAI, 2026a) and Claude Sonnet 4.6 (Anthropic, 2026c) in place of the default

judge (Table A7). CRAFTER scores 50.2 under both Gemini 3.5 Flash and GPT-5.4 and 49.1 under Claude Sonnet 4.6, and its lead over PaperBanana stays between 19.9 and 22.2 points under all three judges. The other rows of Table 2, including those that compare backbones from different families such as GPT-Image-2 against Nano Banana 2, were not rescored, and PaperBanana-Bench keeps its official Gemini 3.1 Pro judge. The CRAFTEDITOR evaluation already averages judges from three model families (Appendix E).

Table A7: CRAFTBENCH overall scores (%) of CRAFTER and PaperBanana, both with Nano Banana 2, under judges from three model families. Gap is CRAFTER minus PaperBanana.

Judge	CRAFTER	PaperBanana	Gap
Gemini 3.5 Flash (default)	50.2	28.0	22.2
GPT-5.4	50.2	30.3	19.9
Claude Sonnet 4.6	49.1	28.7	20.4

G JUDGE PROMPTS

The CRAFTBENCH judge runs as four jobs, one per task type (text-to-image, key-element composition, sketch-conditioned generation, mask-completion). All four share the system scaffold and scoring anchors of the first box below. The per-sample aspect set ({`aspects`}) is drawn from the second box, and the user message is filled from the third. Text blanks {{...}} are filled from sample metadata, and image blanks [[...]] are attached as JPEG.

System prompt and scoring anchors (shared by all four jobs, with the edit jobs additionally foregrounding input_fidelity)

You are a STRICT, skeptical reviewer of scientific figures. You will see the original request given to an illustrator, optionally an INPUT IMAGE they were told to work from, and ONE candidate figure. One of the candidates you review is a real human-made figure from a published paper/poster/blog; the other is AI-generated. AI figures often look glossy yet hide subtle flaws: garbled or fake text, plausible-but-wrong components, generic placeholders, or (for edit tasks) silently ignoring/altering the input image. Hunt for these.

Score each aspect 0-10 with CONSERVATIVE anchors:
- 9-10 = flawless on this aspect after careful search (you found NO weakness)
- 7-8 = good, minor issues
- 5-6 = acceptable but clearly flawed
- 3-4 = major problems
- 0-2 = fails / unrelated / garbled

Do NOT default to 9-10. Most figures have at least minor flaws; if you are about to give 9-10, re-examine and try to find a weakness first. Reward correctness and communication, not surface polish.

Aspects to score for THIS sample:
{aspects}

First list concrete weaknesses you actually observe (be specific; "none" only if truly flawless), then score. Return STRICT JSON only:
{"weaknesses": ["<specific flaw>", ...], "scores": {<aspect keys>}}

Figure A2: CRAFTBENCH judge: shared system prompt and scoring anchors. The edit jobs additionally foreground input fidelity.

Aspect definitions filling the {aspects} slot. Text-to-image uses content_faithfulness, readability, and one format aspect, while the edit tasks use content_faithfulness, readability, and input_fidelity.

content_faithfulness
Does it convey the content in the caption/brief ACCURATELY and COMPLETELY? Check the specific components, relationships and flow named in the caption are present and correct. CRITICAL: scrutinize the text in the figure: are labels REAL, specific and correct, or are some garbled / nonsensical / vague placeholders / AI-gibberish? Are any components fabricated or missing? Plausible-looking but wrong = low. 9-10 only if every caption element is present, correct, and all text is genuine and legible.

readability
Is the information easy to extract: legible text, clear visual flow, sensible grouping, no clutter/overlap/spaghetti arrows? 9-10 only if a reader navigates it effortlessly.

format_academic | format_poster | format_infographic [t2i: one, by content type]
academic = clean single-figure diagram, restrained styling, crisp components/arrows; no poster banner, no infographic decoration.
poster = real conference poster: title banner, author/affiliation line, >=3 sections with headers, rich content; penalize a thin skeleton.
infographic = casual explainer (Distill/Quanta/blog style): illustrative icons or a visual metaphor, narrative callouts, friendly non-academic typography; penalize a dry academic block diagram.

input_fidelity [edit tasks only; judged OUTPUT vs INPUT IMAGE only]
key-element : are the SPECIFIC provided elements actually REUSED (recognizably the same icons/photos/charts), not replaced by similar-but-different ones? Ignoring or substituting them MUST score 0-4; 8-10 only if most/all provided elements are clearly reused.
sketch : scores (a) LAYOUT-FOLLOWING (keep the sketch's panels, labels, arrow connections, reading order) and (b) POLISH (a genuinely refined publication figure, NOT a near-copy of the rough sketch). A near-copy MUST score 0-4 even if the layout matches; 8-10 only if faithful AND clearly polished.
mask-compl. : scores (a) SUBSTANTIVE FILL (the blank region is filled with appropriate content; leaving it empty or reproducing the input MUST score 0-3) and (b) PRESERVATION (the rest is unchanged); 8-10 only if the fill is correct AND the rest is preserved.

Figure A3: CRAFTBENCH judge: aspect definitions that fill the {aspects} slot. Text-to-image uses content faithfulness, readability, and one format aspect; the edit tasks use content faithfulness, readability, and input fidelity.

User message (text blanks filled from sample metadata; image blanks attached as JPEG)

Text-to-image
ORIGINAL REQUEST
Task: t2i | Requested figure style: {{STYLE}}
Target figure caption:
{{CAPTION}}
SOURCE PAPER CONTEXT (judge faithfulness against this):
{{PAPER_CONTEXT}}
Illustrator's brief:
{{INSTRUCTION}}
Candidate figure to score:
[[CANDIDATE_IMAGE]]

Edit tasks (key-element / sketch / mask-completion)
Target figure caption: {{CAPTION}}
SOURCE PAPER CONTEXT (content correctness ONLY; does NOT affect input_fidelity):
{{PAPER_CONTEXT}}
INPUT IMAGE given to the author (a brief to follow/fill, NOT to reproduce):
[[INPUT_IMAGE]]
[mask-completion only] The blanked region should contain: {{REGION_DESCRIPTION}}
Result to score (compare its structure/elements to the INPUT above):
[[CANDIDATE_IMAGE]]

Figure A4: CRAFTBENCH judge: user message template. Text blanks $\{\{\ldots\}\}$ are filled from sample metadata and image blanks $[[\ldots]]$ are attached as JPEG.

H HUMAN EVALUATION

Because the CRAFTBENCH score is produced by a VLM judge, we verify that it reflects *human* preference with a blind pairwise study. Three graduate-level annotators, compensated at a local rate of $25 per hour (above the regional minimum wage), each rate the same random sample of 60 cases spanning all four tasks and three figure types. For each case, an annotator compares the model output against the original human-drawn figure through the web interface of Figure A6, where the two are shown as "Figure A" and "Figure B" in a randomized, hidden order under the same conditioning input, and selects *A is better*, *Tie*, or *B is better*, following the instruction in Figure A5.

Instruction shown to annotators in the blind pairwise study

You are shown the conditioning input the model was given (a caption, and for edit tasks an input image such as a partial figure, an element collage, or a rough sketch) and two rendered figures, A and B. One is the model output and one is the original human-drawn figure, in randomized and hidden order. Choose which figure better satisfies the request, namely faithful to the described content, readable, well-composed, and (for edit tasks) consistent with the provided input image, or Tie. Judge on quality only.

Figure A5: Instruction shown to annotators in the blind pairwise human-evaluation study.

Mapping each comparison to a $\{\text{MODEL}, \text{TIE}, \text{HUMAN}\}$ verdict and comparing it against our automatic judge, the metric agrees with the majority human verdict on 71.7% of cases at Cohen's $\kappa = 0.58$ (Table A8), confirming that the automatic score is a reliable proxy for human judgment across the four tasks and three figure types.

Table A8: Human study (N=60 blind pairwise comparisons on a random sample, three annotators, spanning all four tasks and three figure types). The judge tracks the majority human verdict.

Automatic judge	Agreement (%)	Cohen's κ
CRAFTBENCH judge (ours)	71.7	0.58

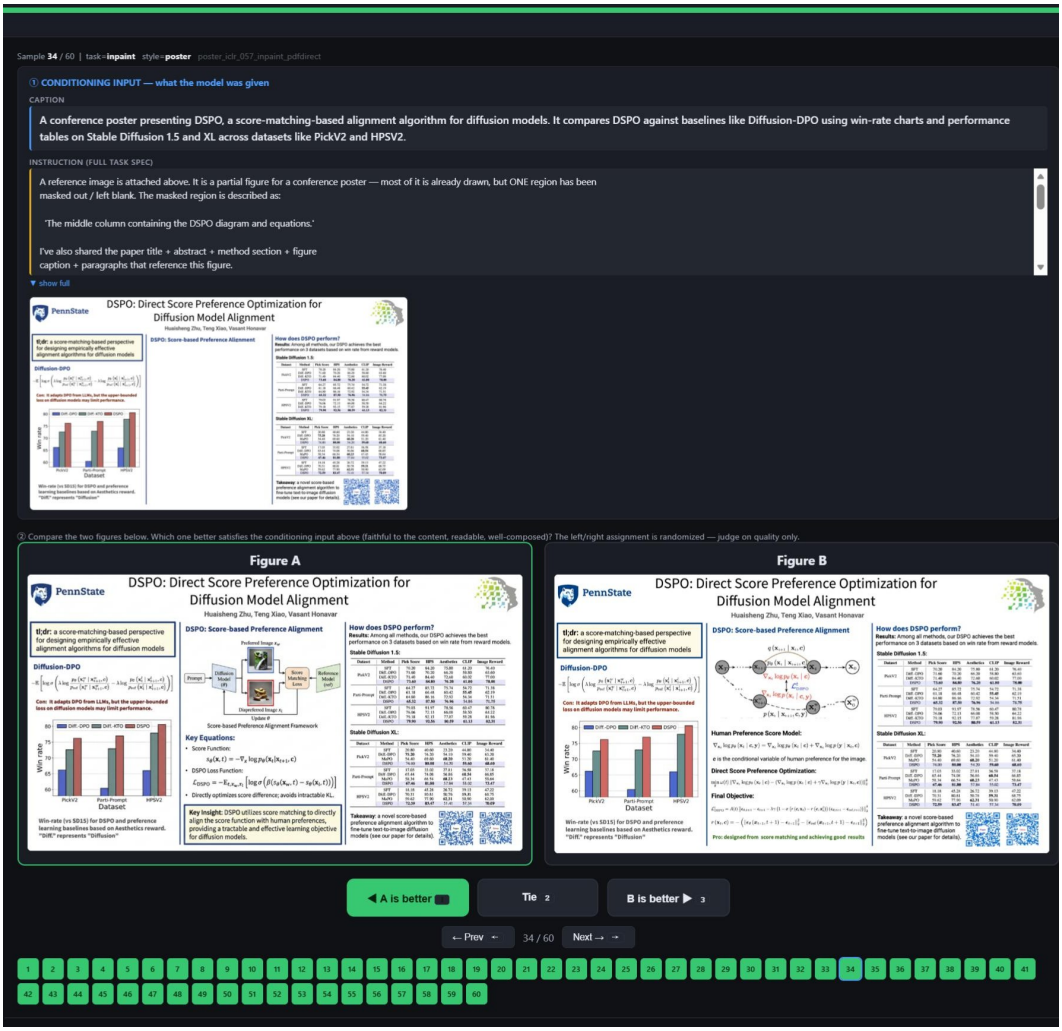


Figure A6: The blind pairwise human-evaluation interface. The conditioning input (caption, full task instruction, and the input image for edit tasks) is shown at the top, and the annotator then chooses which of the two randomized figures better satisfies it.

I HUMAN EDITING STUDY

Table 4 scores how faithfully each conversion reproduces its input raster, but not how much effort a person needs to revise the converted file. We measure this effort directly in a small human editing study. We sample 20 figures stratified across figure types, and Edit-Banana, AutoFigure-Edit, and CRAFTEDITOR each convert every figure with the settings of Appendix A.2. Two further annotators, in addition to the three of Appendix H, independently open each converted output in a standard editor, side by side with the original figure, and edit it until it matches the original, with a cap of 10 minutes per file. Each trial ends in one of three outcomes: *matched*, *editable but unmatched* within the cap, or *uneditable* when the file exposes no editable structure. For each trial we record the outcome, the

time to reach a match, and an editability rating from 1 to 5 that reflects whether elements behave like ordinary slide objects, that is, whether they can be selected, moved, and retyped individually. Trials that do not reach a match are counted at the cap when computing the median time.

As Table A9 shows, CRAFTEDITOR reaches a match in 80% of cases with a median time of 2 minutes, half that of AutoFigure-Edit (65%, 4 minutes), and receives the highest editability rating (4.20 against 3.50 and 1.70). The median time of Edit-Banana sits at the cap, and 70% of its outputs offer no editable structure. The ranking agrees with the fidelity scores of Table 4, where CRAFTEDITOR leads AutoFigure-Edit by the widest margins on the structural axes, text (7.61 vs. 6.04) and arrows (7.83 vs. 6.39). With 20 figures and two annotators, the study cannot resolve small differences between systems, and its numbers should be read as initial evidence rather than precise estimates. The task asks annotators to reproduce the input figure, which measures the cost of repairing a conversion but not revisions beyond reproduction, such as restructuring a layout or restyling a figure.

Table A9: Human editing study on 20 figures with two annotators. Match: share of cases matched within 10 minutes. Time: median minutes to a match, with unmatched trials counted at the cap. Rating: editability from 1 to 5. Uneditable: share of outputs with no editable structure.

System	Match (%)	Time (min)	Rating	Uneditable (%)
Edit-Banana	20.0	10.0	1.70	70.0
AutoFigure-Edit	65.0	4.0	3.50	15.0
CRAFTEDITOR	**80.0**	**2.0**	**4.20**	**5.0**

J LIMITATIONS

Our headline numbers rely on closed-source models for both image-generation backbones (Gemini 3.1 Flash Image, with comparisons to Gemini 3.0 Pro Image and `openai/gpt-image-2`) and evaluation judges (Gemini 3.1 Pro for PaperBanana-Bench, Gemini 3.5 Flash for CRAFTBENCH), making the harness contribution conditional on proprietary-model access and judge biases; CRAFTER and CRAFTEDITOR further rely on a strong language model treated as a black box. Per-sample latency and cost are non-trivial: a single CRAFTER run executes up to three parallel image generations followed by up to three refinement rounds, and CRAFTEDITOR adds roughly four agentic VLM rounds plus an SVG composition step, so deploying the harness at scale requires the corresponding API budget and wall-clock time. Our baselines are published research systems, and we do not compare against interactive commercial design assistants under a matched budget. Finally, CRAFTBENCH passes manual quality assurance on every editing-task sample, but the benchmark remains small relative to training corpora and leans toward academic and poster figures, with infographics the smallest pool; its reference-conditioned subsets hold only 30 to 40 samples each, so their scores carry wider uncertainty, and broadening infographic coverage is the natural next iteration.

K CASE STUDIES

This appendix collects the qualitative case studies referenced from the main text. Figure A7 compares CRAFTEDITOR against prior raster-to-editable systems on five CRAFTBENCH outputs; per-panel scores are the three-VLM judge mean.

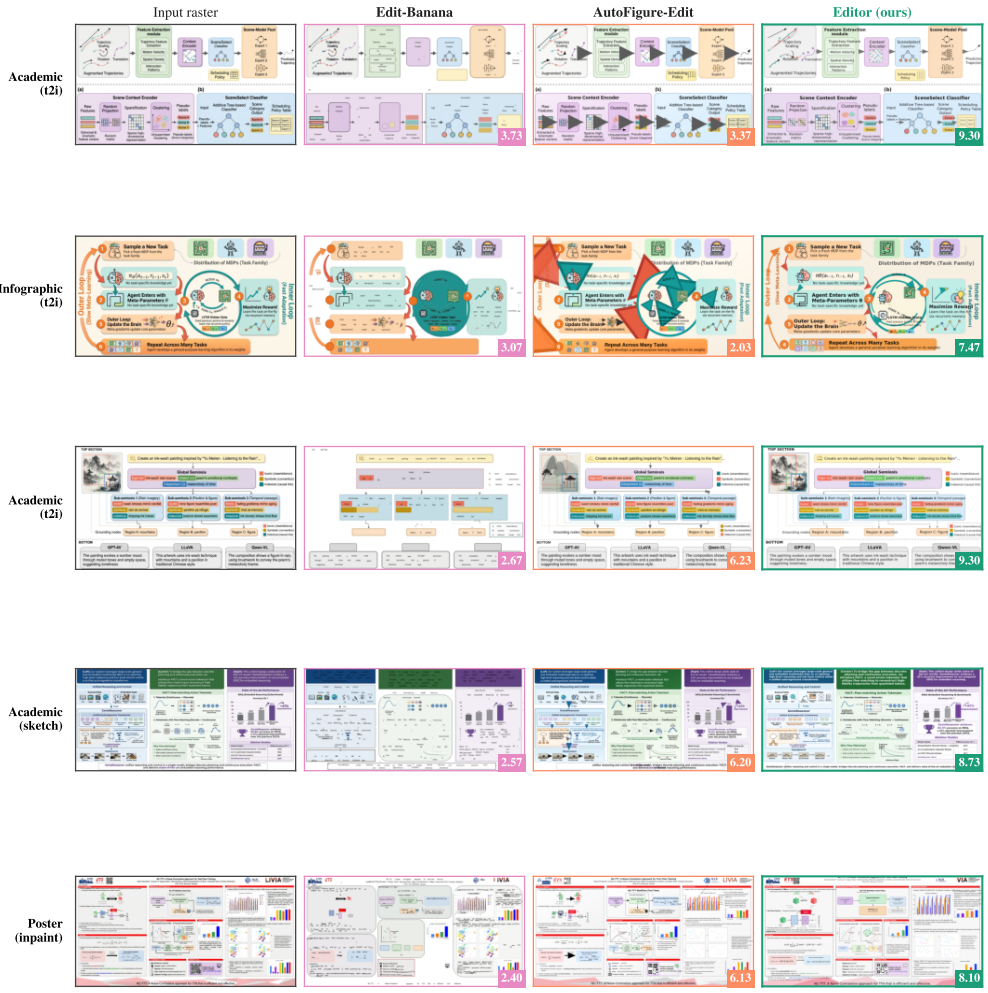


Figure A7: Qualitative comparison of editable-output systems on five cases. Columns: input raster, Edit-Banana, AutoFigure-Edit, CRAFTEDITOR (green frame); numbers are three-VLM judge means.

CRAFTER on input-honoring editing tasks. On the CRAFTBENCH editing samples shown in Figure 5, the baselines visibly regenerate from the caption and ignore the conditioning input, so the input-fidelity aspect scores them far below CRAFTER. The conditioning-input columns and the CRAFTER columns share spatial structure, while the baseline columns share content with neither.

CRAFTER on multi-component academic figures. On academic samples that combine multiple paper-named components and complex flow (e.g., a manipulation-order framework or a mesh-movement network), the baselines miss the specific paper-named components and substitute generic stock visuals; CRAFTER's verify-then-refine loop catches the missing component on the first round, and the structured corrective layer adds a layout note that pins the component to the correct sub-region for the second round.

L FAILURE CASES

CRAFTER does not win on every sample. Figure A8 collects three cases on which the automatic judge prefers the human-drawn target, each shown with its conditioning input and ground truth. Each row isolates one failure mode and the harness stage responsible for it.

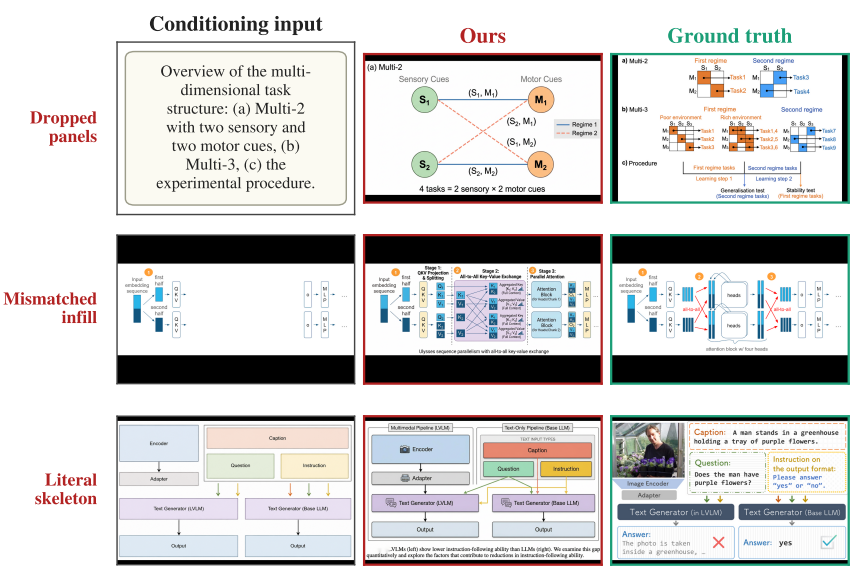


Figure A8: Representative CRAFTER failure cases. Columns: conditioning input (caption for the text-to-image row, reference image for the reference-conditioned rows), CRAFTER (red frame), ground truth (green frame). Rows, top to bottom: dropped panels, mismatched infill, literal skeleton.

- **Dropped panels** (text-to-image). The intent reasoner collapses a multi-panel caption (a/b/c) into a single panel; the missing panels never enter the shared specification, so the verify-then-refine loop cannot recover them.
- **Mismatched infill** (mask-completion). CRAFTER regenerates the masked region in a clashing boxed register that no longer continues the surrounding diagram preserved from the input, breaking visual continuity at the mask boundary.
- **Literal skeleton** (sketch-conditioned). The layout follows the sketch faithfully but stays an abstract block diagram, omitting the concrete illustrative example (a photo and a worked question/answer) that the target uses to convey the point.

Dropped panels trace to intent reasoning, mismatched infill to a critic that scores structure but not whether infilled content stays faithful to the input, and literal skeletons to the backbone. The first two point to concrete fixes: a panel-count check and a mask-boundary continuity check in the critic.