FIGSTRUCT: OBJECT HIERARCHIES IN THE LOOP FOR AGENTIC SCIENTIFIC FIGURE REDESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Improving the visual quality of scientific figures often requires time-consuming manual editing. Researchers must adjust layouts, labels, and connections while preserving the underlying scientific meaning. Image generation can improve presentation, but further revisions require an editable representation of the result. We introduce FIGSTRUCT, an agentic framework that couples visual redesign with object-hierarchy-guided SVG correction. A local open-source controller analyzes the input, retrieves design references, and plans a revised composition to guide an image generator. The generated design is converted to SVG and organized into an object hierarchy. Enclosing groups provide diagnostic context, while object bindings identify the SVG elements selected for revision. These bindings are updated after structural changes to support subsequent corrections. Correction rounds run locally, followed by one paid model call for final global refinement. On 150 scientific figures, our best redesign configuration achieves the highest mean scores across all five model-judged visual-quality dimensions among the compared methods. With the generator fixed, agentic guidance improves all five scores for GPT Image 2.0 and four for Qwen Image 2.0 Pro over direct prompting. Reconstructed SVGs attain 99.39% semantic text preservation against the redesigned image and 96.34% strict native-text editability. On 50 complex figures, manual post-editing to a common design target averages 9 minutes 15 seconds for FigStruct outputs, compared with 45 minutes 19 seconds for AutoFigure-Edit outputs. Including retries, mean paid API cost for SVG reconstruction and refinement is $0.009429 per output. This cost is 96.78% lower than that of AutoFigure-Edit and 99.61% lower than that of CraftEditor.

1 INTRODUCTION

Information-dense diagrams communicate through the arrangement of labels, graphical elements, and connections (Larkin & Simon, 1987). Redesigning them requires improving this arrangement while preserving the relationships it conveys. Scientific figures exemplify this setting through their depiction of complex methods and relationships (Rougier et al., 2014; Agrawala et al., 2011). Even the number of connector crossings affects comprehension (Purchase, 1997). A draft figure can express the intended content (Belouadi et al., 2024b; Tan et al., 2025), but improving its layout takes effort and must preserve editability (Sun et al., 2026).

Generative systems automate scientific illustration through planning and rendering (Zhu et al., 2026a;b; Li et al., 2026; Huang et al., 2026; Mondal et al., 2024). Reconstruction methods recover editable graphics from images (Rodriguez et al., 2025; He et al., 2026; Lin et al., 2026). Scientific image generation can misrepresent object attributes and relationships, motivating explicit content requirements when planning a new composition (Zhang et al., 2025). Reconstruction then requires comparing the resulting design with the rendered SVG and revising code to correct discrepancies (Lin et al., 2025). Each correction must identify the editable elements responsible for the visual error. Repeated edits can alter correct regions or lose earlier requirements, making the scope of each correction important (Zhang et al., 2026; Wu et al., 2026).

Structured generation methods such as FigTree and Crafter organize construction and verification through recursion and evolving specifications (Liu et al., 2026; Zhao et al., 2026). Correcting an existing SVG requires context beyond the affected objects, yet revising that entire context can disturb

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.

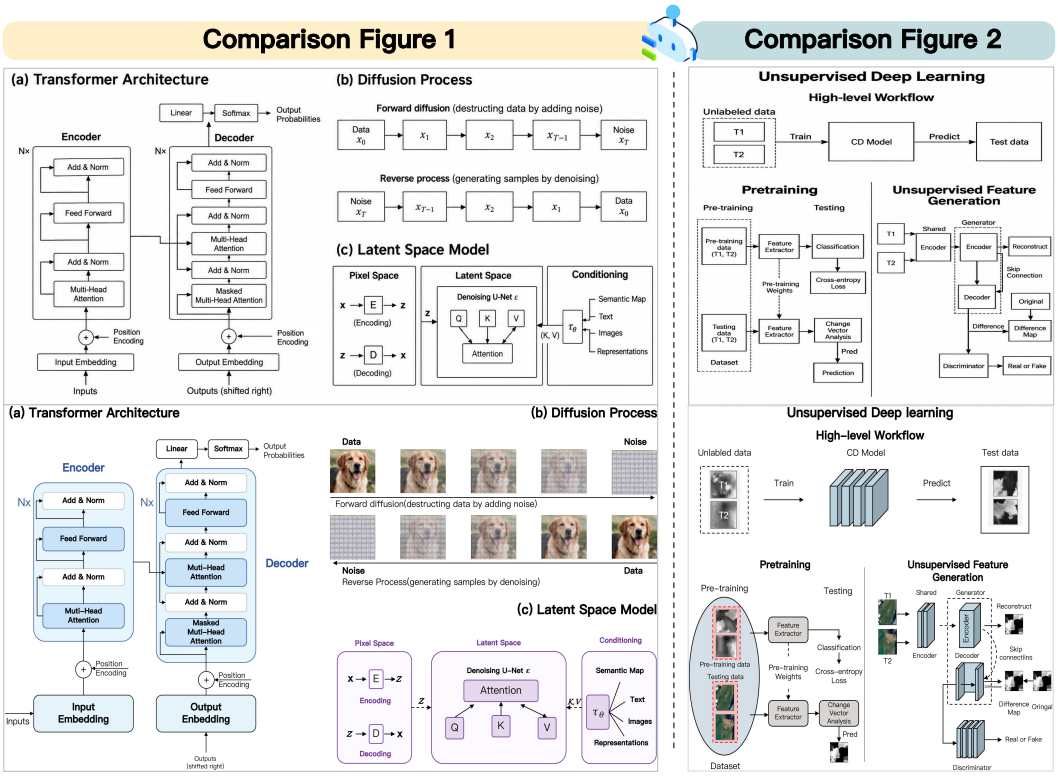


Figure 1: Scientific figure redesign with FigStruct. In each example, the input figure appears above the rendered final SVG, showing changes in layout, grouping, and graphical elements.

correct elements. We separate these scopes through an object hierarchy: enclosing groups provide diagnostic context, while object bindings identify the SVG elements to revise. Updating these bindings after structural changes allows subsequent corrections to address the revised figure.

We introduce FIGSTRUCT, an agentic framework that couples visual redesign with hierarchy-guided SVG correction. A local open-source controller analyzes the input figure, retrieves design references, and plans a composition for an image generator. The controller organizes an initial SVG into an object hierarchy and compares its render with the generated design image. It corrects regions in parallel, then reviews the merged result. Local correction rounds precede one paid global refinement pass for cross-region consistency. Figure 1 illustrates the resulting editable redesigns.

We evaluate visual redesign and editable reconstruction on 150 scientific figures. Our best configuration leads all five model-judged visual-quality dimensions among the compared methods. With the generator fixed, agentic guidance improves all five scores for GPT Image 2.0 and four for Qwen Image 2.0 Pro over direct prompting. Reconstructed SVGs attain 99.39% semantic text preservation against the redesigned image and 96.34% strict native-text editability. Including retries, paid API cost for reconstruction and refinement averages $0.009429 per output, 96.78% below AutoFigure-Edit and 99.61% below CraftEditor. On 50 complex figures, manual post-editing to a common design target averages 9 minutes 15 seconds for FigStruct outputs generated with the local controller. The mean for AutoFigure-Edit outputs is 45 minutes 19 seconds. Our contributions are:

- We introduce FIGSTRUCT, combining visual redesign and editable SVG output with a local open-source controller for planning and iterative correction.
- We develop hierarchy-guided SVG correction that separates diagnostic context from intended edits and updates object bindings after structural changes.
- We construct 150 pairs of input and reference figures for planned public release and evaluate visual quality, fidelity, editability, API expenditure, and manual post-editing effort.

2 Related Work

Scientific figure generation and redesign. Scientific figures can be generated from declarative specifications through constrained optimization, as in Penrose (Ye et al., 2020), or from documents through agentic planning and refinement (Mondal et al., 2024; Zhu et al., 2026a;b). For editable output, SciFig combines hierarchical layouts with native components and generated assets, while DiagramAgent generates and checks diagram code (Huang et al., 2026; Wei et al., 2025). Working from existing visual content introduces a complementary task: interpreting what to retain while changing its presentation. SketchAgent structures hand-drawn sketches, and SciDiagramEdit learns instruction-guided vector edits from paper revisions (Tan et al., 2025; Sun et al., 2026). PaperBanana-Interact uses human feedback and internal critique to address quality drift and forgotten requirements during refinement (Wu et al., 2026). FigStruct combines visual reorganization and editable reconstruction, guided by the scientific content of the input figure.

Editable graphics generation and reconstruction. Graphics programs expose elements that can be edited after generation. Learned representations support hierarchical vector generation in DeepSVG (Carlier et al., 2020), TikZ synthesis from text or images (Belouadi et al., 2024a;b), and SVG generation from multimodal inputs (Rodriguez et al., 2025; Yang et al., 2026; He et al., 2026; Xing et al., 2025). Another line of work optimizes vector graphics through differentiable rendering (Li et al., 2020; Ma et al., 2022) or diffusion guidance (Jain et al., 2023; Xing et al., 2024; Wu et al., 2025). For complex scientific figures, hybrid reconstruction combines native SVG elements with extracted image assets. AutoFigure-Edit aligns SVG placeholders with indexed components and preserves their mappings during refinement (Lin et al., 2026). CraftEditor verifies asset extraction and iteratively refines SVG composition (Zhao et al., 2026). FigStruct also uses hybrid SVGs, with an object hierarchy that links visual errors to the elements selected for correction.

Structured representations for iterative correction. Iterative correction requires feedback and a representation on which to act. Self-Refine and Reflexion use generated feedback and stored verbal reflections (Madaan et al., 2023; Shinn et al., 2023). Visual reasoning systems make intermediate steps explicit through modular programs (Gupta & Kembhavi, 2023; Surís et al., 2023) or drawn visual aids (Hu et al., 2024), while VCoder uses visual-tool cues to guide SVG revision (Lin et al., 2025). For scientific figures, Crafter accumulates typed revisions in an evolving specification. FigTree recursively constructs SVG fragments with boundary ports and uses code and rendered previews for localized repair (Liu et al., 2026). These systems connect feedback to structured generation and editing. FigStruct centers correction on an object hierarchy derived from the current SVG. The hierarchy provides group-level context for diagnosis while restricting each edit to selected objects. After structural changes, reconciliation updates object membership and SVG bindings so that subsequent corrections operate on the revised figure.

3 Method

3.1 Task and Overview

Given an input scientific figure I_{src}, FigStruct produces a redesigned SVG $S^{\star}$ using the source labels, entities, and connections as content constraints. The source image is the only user-provided input. The output combines native SVG text, shapes, and connectors with individually selectable raster assets for complex visual content.

Figure 2 summarizes source-grounded redesign, hierarchy-guided SVG correction, and Global Refinement. A local open-source controller plans the redesign with retrieved references and iteratively corrects its SVG before cross-region refinement.

3.2 Source-Grounded Visual Redesign

Visual redesign changes presentation while preserving scientific content. In Figure Understanding, the controller identifies labels, entities, containment relations, and scientific connections, requesting higher-resolution crops for ambiguous regions. This analysis records the content to preserve when

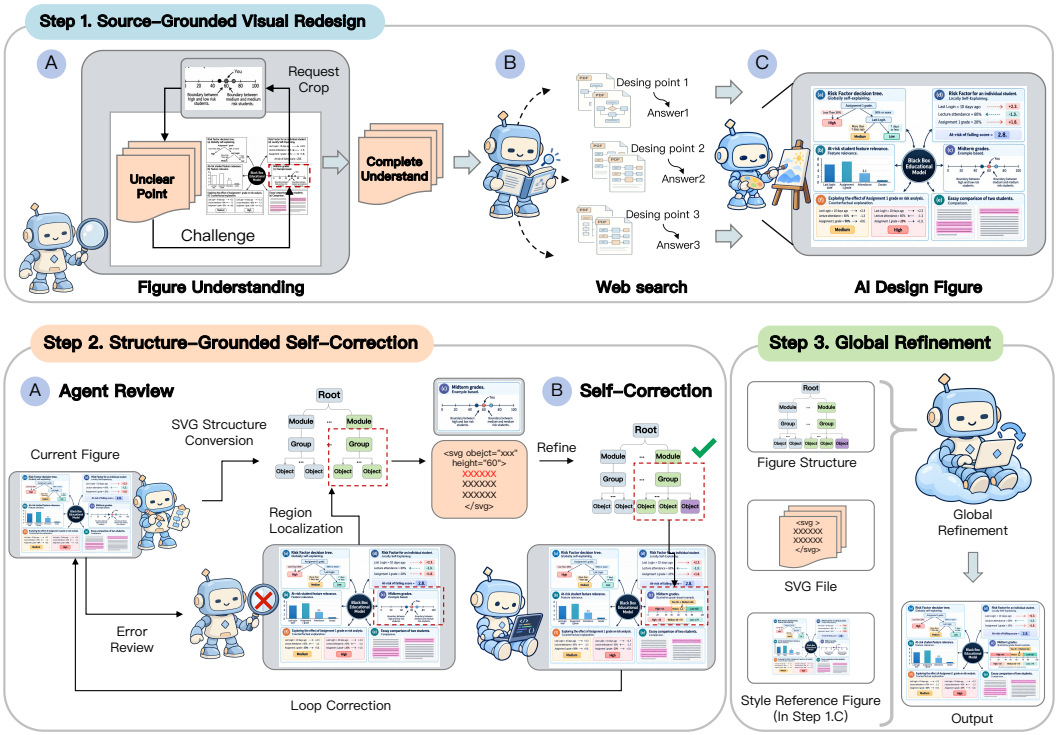


Figure 2: Overview of FigStruct. Step 1 interprets the source and guides visual redesign with retrieved references. Step 2 begins with initial SVG generation and one-time object-tree construction, omitted from the diagram. Correction rounds use and update this hierarchy to localize errors, repair regions in parallel, and review the merged SVG. Step 3 jointly refines the SVG using its object hierarchy and the fixed design image.

reorganizing the figure. During Web Search, the controller retrieves design references to address weak grouping, excessive density, and ambiguous connector routing.

Retrieved examples inform a design brief on grouping, spacing, and visual conventions. The controller combines this brief with source content to plan canvas geometry, module proportions, reading order, and connectors. It reviews the composition before prompting a frozen image generator. The resulting design image I_{AI}, labeled AI Design Figure in Figure 2, provides the content and visual reference for subsequent SVG reconstruction and correction.

3.3 Structure-Grounded Self-Correction

Diagnosing a visual error can require group-level context even when only a few objects need revision. The object hierarchy separates this context from the edit scope and maintains object bindings as the SVG changes. We first generate an initial SVG S_0 from I_{AI}, using native text and graphical elements together with selectable raster assets where needed. SVG Structure Conversion organizes these existing elements into an object hierarchy once. Subsequent iterations revise the SVG and update the same hierarchy.

SVG Structure Conversion. The hierarchy $\mathcal{T}_t$ organizes the current SVG S_t into modules, nested groups, and individual objects under a figure root. Each node records content, spatial region, and bindings to SVG elements or raster assets. The object-tree JSON file is labeled Figure Structure in Figure 2. We call a module or subregion selected for joint revision a *target*. Parent groups supply context, and bindings identify editable code.

Agent Review. At each outer-loop iteration t, Error Review compares the rendered SVG $\mathcal{R}(S_t)$ with I_{AI} to identify content and presentation errors. Content checks identify missing or duplicated

labels, incorrect entities, and erroneous connections. Presentation checks examine overlap, clipping, spacing, and alignment. Region Localization maps each diagnosis to the affected targets and objects in $\mathcal{T}_t$. Each edit plan records the required correction, selected object identifiers, and enclosing groups needed to interpret the error.

Self-Correction. Each parallel worker receives the complete SVG fragment rooted at its target as context. The correction plan defines the permitted edit scope. Object-focused repairs modify only selected objects, whereas target-level revisions allow coordinated changes to layout, object organization, and internal connections within the target. In both cases, the worker returns a complete replacement fragment with the same target identifier. Unselected target subtrees and protected asset identities and content are preserved.

All native-SVG repairs remain within the fixed rendered boundary B_T, measured when the worker begins local repair. Each candidate is rendered for review and checked for boundary overflow, with local repair continuing until accepted.

Loop Correction. A structural edit can change group membership, introduce objects, or remove them. After the parallel workers finish, accepted target replacements Δ_t update the SVG and object records together:

$$\begin{aligned} S_{t+1} &= \mathrm{Merge}\left(S_t, \Delta_t\right), \\ \mathcal{T}_{t+1} &= \mathrm{Reconcile}\left(\mathcal{T}_t, S_{t+1}\right). \end{aligned} \tag{1}$$

Merge inserts the selected fragments while retaining unselected subtrees and protected assets. Reconcile updates membership and SVG bindings, retaining existing identifiers and registering new objects. Objects absent after merging are checked against I_{AI} and the current render to distinguish justified removal from loss of required content. The next review uses these updated groups and bindings to locate errors in the revised SVG. Whole-figure review then checks the merged render and identifies corrections for the next round. Step 2 performs at most two correction rounds, with no fixed limit on local errors addressed per round. Nested-target merging, cross-module connector handling, and execution traces appear in Appendix G.

3.4 Global Refinement

Typography, inter-module spacing, and use of canvas space require coordination beyond local repairs. Step 3 receives the corrected SVG S_K, its corresponding object-tree JSON, and the fixed design image I_{AI} from Step 1.C as the style reference. Parent, child, and sibling relationships guide a high-capacity refiner in revising grouping and canvas geometry across target boundaries in one pass. The refiner is instructed to retain content, assets, connector validity, and the design aspect ratio. The final SVG $S^\star$ is checked for successful XML parsing and rendering.

4 Experiments

We evaluate visual redesign and SVG reconstruction through five questions. **RQ1:** Does agentic guidance improve visual quality over direct prompting? **RQ2:** Can reconstruction preserve content and appearance while recovering editable elements? **RQ3:** How do API cost, runtime, and final SVG quality vary across controllers? **RQ4:** What does each refinement component contribute? **RQ5:** How does FigStruct affect element-level and relational fidelity across controllers?

4.1 Experimental Setup

Dataset. The benchmark contains 150 pairs of published diagrams and simplified manual redrawings. Fifty participants produced the redrawings, preserving text and logical relations. Retrieved PDFs matching source-paper titles are excluded, as detailed in Appendix A. All methods are evaluated on the same 150 benchmark samples, using the inputs specified for each evaluation setting below. Failed attempts are retried until a final output is obtained for each sample.

Baselines. For visual redesign, we compare FigStruct with PaperBanana (Zhu et al., 2026a) and direct generation using Qwen Image 2.0 Pro or GPT Image 2.0. Direct generation uses the same

source and generator as FigStruct but omits analysis, retrieval, and planning. SVG baselines include VTracer for contour tracing (VisionCortex, n.d.), VFIG-4B for image-to-SVG conversion (He et al., 2026), and AutoFigure-Edit and CraftEditor for agentic reconstruction (Lin et al., 2026; Zhao et al., 2026). We use official implementations and follow the provided instructions, with model settings in Appendix C. AutoFigure and Crafter denote AutoFigure-Edit and CraftEditor, respectively.

Evaluation protocol. Visual redesign starts from the original input figures. Reconstruction methods share the redesigned input from Figure 2, Step 1.C, also used to evaluate rendered similarity and text preservation. Ablations and controller comparisons start from the same original figures and measure final SVG fidelity against those inputs. Full FigStruct outputs are evaluated after Step 3.

Evaluation metrics. GPT-5.6 Sol with high reasoning effort evaluates redesigned images without method or generating-model identities. It rates clarity, logical flow, aesthetic quality, visual expressiveness, and professional polish on a ten-point scale. Scores are averaged over three independent evaluations per image pair. Logical flow assesses visual organization, while source-based fidelity assesses scientific relations. Reconstruction is assessed through SSIM, DINO, LPIPS, text preservation, strict native-text editability, and geometry metrics for canvas fit, overflow, containment, padding, and overlap. Strict native-text editability requires text and semantic symbols to use native SVG text elements, excluding outlines and rasterized text. For component and controller analyses, composite quality averages source-based fidelity, final SVG aesthetic quality, and SVG editability on a 0–100 scale. Fidelity uses manually reviewed item annotations, and the aesthetic component averages seven visual dimensions. SVG editability combines native-text matching and object separability, as defined in Appendix B. Main-text API cost and runtime cover reconstruction and refinement, corresponding to Steps 2 and 3 for FigStruct. Both include all retries through final output, and runtime includes reconnection waits. API cost excludes visual redesign and local compute. Appendix C provides cost breakdowns and failure counts.

Implementation details. The default controller is Gemma-4 31B, deployed on two 80 GB NVIDIA H100 GPUs with temperature 0.3 and a maximum output length of 40,000 tokens. Qwen Image 2.0 Pro or GPT Image 2.0 generates one raster design per input figure. SAM 3.1 extracts raster assets. Step 2 permits two outer correction rounds with three parallel region workers. GPT-5.6 Luna performs one final Global Refinement pass.

4.2 RQ1: Visual Redesign Quality

Both FigStruct configurations improve all five visual-quality dimensions over the input figures in Table 1. With GPT Image 2.0, FigStruct obtains the highest mean scores among the compared methods. Aesthetic quality rises from 4.16 to 8.93, a gain of 4.77 points. Its largest margins over PaperBanana are in logical flow and visual expressiveness, at 0.80 and 1.37 points.

With the image generator fixed, FigStruct improves all five scores for GPT Image 2.0 and four for Qwen Image 2.0 Pro relative to direct prompting. Visual expressiveness increases by 0.59 and 0.89 points, respectively. These matched comparisons support the value of the analysis, retrieval, and

Table 1: Visual redesign quality on 150 scientific figures. Entries are mean model-judged scores on a ten-point scale, with gains over the original figures in parentheses. Higher scores are better, with the highest mean in each column shown in bold.

Method	Image generator	Clarity	Log. Flow	Aes. Qual.	Vis. Expr.	Polish
Original	—	5.10	7.39	4.16	5.30	4.28
Direct	Qwen Image 2.0 Pro	7.82 (+2.72)	7.93 (+0.54)	7.35 (+3.19)	6.75 (+1.45)	7.48 (+3.20)
Direct	GPT Image 2.0	8.70 (+3.60)	8.45 (+1.06)	8.67 (+4.51)	8.39 (+3.09)	8.81 (+4.53)
PaperBanana	Nano Banana Pro	8.89 (+3.79)	8.28 (+0.89)	8.58 (+4.42)	7.61 (+2.31)	8.68 (+4.40)
FigStruct	Qwen Image 2.0 Pro	8.14 (+3.04)	7.77 (+0.38)	7.98 (+3.82)	7.64 (+2.34)	8.07 (+3.79)
FigStruct	GPT Image 2.0	**8.97** (+3.87)	**9.08** (+1.69)	**8.93** (+4.77)	**8.98** (+3.68)	**9.06** (+4.78)

Table 2: SVG reconstruction fidelity, editability, and rendered geometry. All methods reconstruct the same 150 design images from Figure 2, Step 1.C. Bold and underlined values indicate the best and second-best results among non-tracing methods, excluding VTracer.

(a) Reconstruction Fidelity and Editability

Method	Visual Fidelity			Text Fidelity		Editability & Structure			
	SSIM ↑	DINO ↑	LPIPS ↓	Sem. ↑	Layout ↑	Text Edit. ↑	Clean ↑	Raster area	Path Cmds ↓
VTracer (Tracing)	0.9818	0.9234	0.0221	100.00%	100.00%	0.00%	0.0000	0.00%	111686.76
VFIG-4B	0.5885	0.6101	0.4983	76.22%	55.52%	**100.00%**	**0.7961**	0.39%	**86.44**
AutoFigure	0.6650	0.7799	0.3085	98.48%	98.78%	97.56%	0.6596	1.86%	90.66
Crafter	0.6977	**0.8318**	0.2368	98.58%	99.09%	97.26%	0.6639	4.35%	90.56
FigStruct	**0.7531**	0.8288	**0.2129**	**99.39%**	**99.63%**	96.34%	0.5481	4.99%	109.30

(b) Rendered Geometry

Method	Canvas Geometry		Text Geometry			
	GFR ↑	OAR ↓	TBR ↑	TPVR-6px ↓	TPVR-0.20em ↓	Overlap ↓
VTracer (Tracing)	0.8200	0.00003127	-	-	-	-
VFIG-4B	0.0333	0.1091	0.2391	0.8728	0.8562	0.5151
AutoFigure	0.9000	0.002391	0.7932	0.5474	0.3621	0.03445
Crafter	0.8333	**0.0004231**	0.8821	0.5219	0.3013	0.02528
FigStruct	**0.9067**	0.001232	**0.9441**	**0.2723**	**0.1267**	**0.004588**

planning pipeline beyond generator choice. The benefit varies by criterion, as logical flow decreases by 0.16 points with Qwen despite higher visual expressiveness.

4.3 RQ2: Reconstruction Fidelity and Editability

Visual and text fidelity. Among non-tracing methods in Table 2, FigStruct achieves the highest SSIM at 0.7531 and the lowest LPIPS at 0.2129. Relative to Crafter, SSIM increases by 0.0554 and LPIPS decreases by 0.0239. DINO is 0.0030 below the best result. Against the shared Step 1.C reference, FigStruct leads non-tracing methods in semantic text preservation at 99.39% and text-layout correctness at 99.63%. The latter measures assignment to the correct semantic group, so changes in text position or scale do not count as errors.

Text geometry. FigStruct obtains the best results in text containment, both padding-violation measures, and text overlap. Relative to Crafter, TBR increases by 6.20 percentage points, while TPVR-6px, TPVR-0.20em, and text overlap decrease by 47.8%, 57.9%, and 81.9%, respectively. GFR is 0.9067, compared with 0.9000 for AutoFigure, while OAR is 0.001232, second among non-tracing methods. Figure 3 contrasts text placement, connector routing, and module organization across reconstructed workflow and taxonomy diagrams.

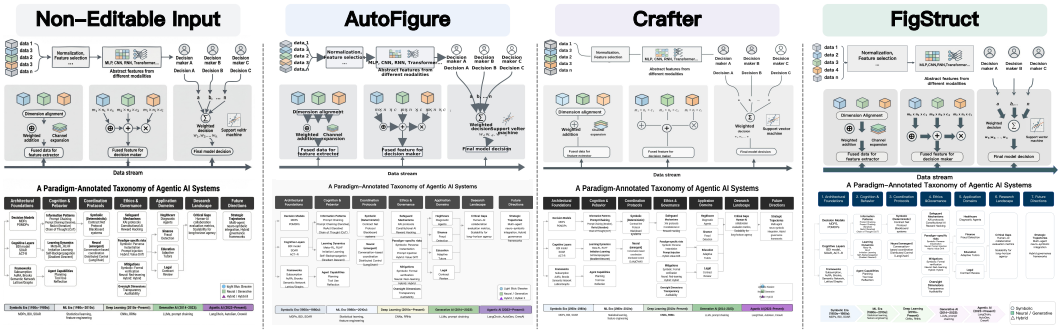


Figure 3: SVG reconstruction of two AI-generated scientific figures. Each row compares an AI-generated raster reference with the corresponding SVG renderings produced by AutoFigure, Crafter, and FigStruct, highlighting differences in the preservation of text labels, connections, and grouping.

Native-text editability. FigStruct reaches a strict native-text editability rate of 96.34%, below AutoFigure and Crafter by 1.22 and 0.92 percentage points, respectively. VTracer obtains the highest rendered similarity but no native-text editability, whereas VFIG-4B reaches 100% native-text editability with a semantic text preservation score of 76.22%. Editability counts visible output text, whereas semantic preservation counts reference items. High editability can therefore coexist with missing labels. Clean measures the fraction of basic shapes and connectors among counted geometric elements. Raster area measures raster coverage, and path-command count describes SVG encoding complexity. These metrics distinguish visual fidelity from representation simplicity.

Manual post-editing effort. On the 50 most complex figures, 400 participants edit outputs from eight methods in PowerPoint to match a common GPT-generated design reference. Each participant edits one output. Each method receives 25 frequent and 25 infrequent PowerPoint users, randomly assigned to its 50 figures. Completion requires unanimous approval from three reviewers, with no time limit. Mean post-editing time is 9 minutes 15 seconds for FigStruct outputs generated with Gemma, compared with 45 minutes 19 seconds for AutoFigure and 58 minutes 10 seconds for Crafter. Editing time reflects both initial output quality and ease of editing. Appendix D provides the protocol and all results.

4.4 RQ3: API Cost and Runtime

Paid API expenditure. With Gemma-4 31B, Steps 2 and 3 average 90.14 successful local responses and one paid API attempt per output in Table 3. API expenditure is $0.009429 per output, 96.78% lower than that of AutoFigure and 99.61% lower than that of Crafter.

Runtime. With three region workers, Gemma averages 29.45 minutes per output, compared with 5.73 for AutoFigure and 27.43 for Crafter. This configuration requires local hardware and has a longer reconstruction time than both baselines.

Controller trade-offs. FigStruct leads composite quality in Figure 4: Gemma-4 31B scores 85.47, GLM-5.3 Flash 87.10, and GPT-5.6 Terra 88.34.

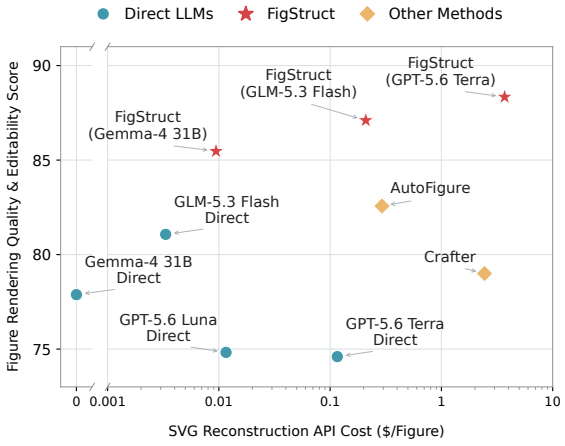


Figure 4: Composite quality versus paid API cost for reconstruction and refinement, covering Steps 2 and 3 for FigStruct. Quality averages source-based fidelity, aesthetic quality, and SVG editability. Positive costs use a log scale, with zero shown separately.

Gemma exceeds AutoFigure and Crafter by 2.90 and 6.47 points at the lowest API cost among FigStruct variants. All three variants are Pareto-optimal in API cost and quality among the evaluated

Table 3: Mean cost and runtime over 150 outputs, including all API retries and reconnection waits. FigStruct uses Gemma-4 31B, with three region workers in Step 2. API cost excludes visual redesign and local compute. Local counts include successful responses.

Metric	AutoFigure	Crafter	FigStruct
API attempts ↓	1.060	58.45	1.000
Local responses	–	–	90.14
Runtime (min) ↓	5.733	27.43	29.45
API cost (USD) ↓	0.2924	2.439	**0.009429**

Table 4: Full-workflow ablations on 150 input figures with Gemma-4 31B. Component names follow Figure 2. Scores use a 0–100 scale, with higher values preferred. Bold and underlined values indicate the best and second-best results.

Method	Fidelity ↑	Aesthetic quality ↑	SVG editability ↑
FigStruct	**95.85**	**87.75**	**72.81**
w/o Global Refinement	82.84	75.20	69.91
w/o Region Localization	86.61	76.57	69.56
w/o SVG Structure Conversion	<u>89.53</u>	<u>77.12</u>	<u>71.97</u>

configurations. Against matched direct generation, Gemma, GLM, and Terra gain 7.59, 6.03, and 13.74 points, respectively.

4.5 RQ4: Component Ablations

Table 4 reports ablations of SVG Structure Conversion, Region Localization, and Global Refinement from the full Gemma-4 31B workflow in Figure 2. All variants use the same inputs, model settings, and iteration limits. Each ablation removes a component and its associated calls, without reallocating unused budgets to the remaining correction stages.

SVG Structure Conversion. Without the hierarchy, the controller queries and edits the complete SVG code. Fidelity falls by 6.32 points, aesthetic quality by 10.63, and editability by 0.84. The larger fidelity and aesthetic losses link the hierarchy to content preservation and visual organization during correction.

Region Localization. Without Region Localization, fidelity, aesthetic quality, and SVG editability fall by 9.24, 11.18, and 3.25 points, respectively. Losses exceed those from removing the hierarchy on all three measures, supporting explicit object selection during correction.

Global Refinement. Omitting Step 3 yields the largest fidelity and aesthetic losses, at 13.01 and 12.55 points, alongside a 2.90-point editability loss. Global refinement therefore remains important after local repair, especially for content fidelity and overall presentation.

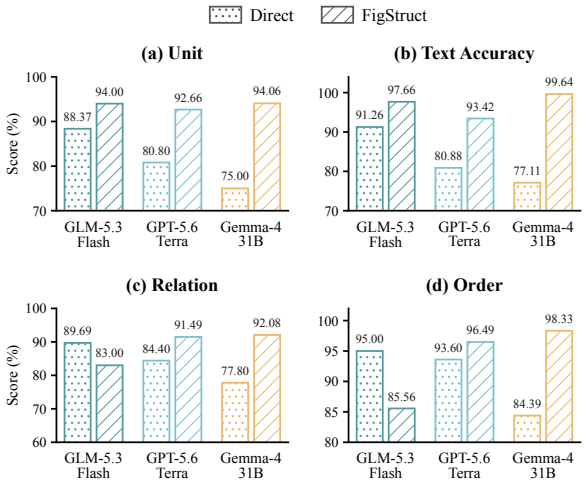


Figure 5: Direct SVG generation versus full-workflow FigStruct with matched controllers. Both use the same 150 original figures as inputs and fidelity references. Higher scores are better, and panel scales differ.

4.6 RQ5: Effect of Controller Choice

Gemma-4 31B ranks lowest under direct generation but highest with FigStruct on all four fidelity measures in Figure 5. Its gains are 19.06 percentage points in unit preservation, 22.53 in text accuracy, 14.28 in relation preservation, and 13.94 in reading order. GPT-5.6 Terra improves all four measures with smaller gains. Controller rankings therefore depend on the correction workflow.

GLM-5.3 Flash gains 5.63 percentage points in unit preservation and 6.40 in text accuracy. Relation preservation and reading order fall by 6.69 and 9.44 points, respectively, despite the 6.03-point composite gain in RQ3. The composite gain thus masks relational errors, as detailed in Appendix C.3.

Gemma leads source-based fidelity, whereas GPT-5.6 Terra leads composite quality. Controller choice therefore depends on which objective takes priority.

5 Conclusion

We presented FigStruct, an agentic framework for scientific figure redesign with editable SVG output. An object hierarchy links visual diagnoses to SVG elements, separates diagnostic context from edit scope, and tracks structural changes across correction rounds. Whole-figure review checks merged parallel repairs before global refinement. Experiments on 150 figures show gains in visual quality, content preservation, and editable reconstruction with low paid API expenditure. Ablations support all three correction components, and a 400-participant study finds shorter mean post-editing time on 50 complex figures. Further work should reduce correction latency and test naturally occurring research drafts and other information-dense diagrams.

AI USE STATEMENT

AI tools assisted manuscript preparation and language polishing. During development of FigStruct, AI assisted code implementation and debugging under author supervision. The authors reviewed and verified the AI-assisted text and code, checked experimental outputs, and interpreted the results. The authors take full responsibility for the paper.

REPRODUCIBILITY STATEMENT

Sections 3 and 4 describe the framework and experimental settings, with implementation details and correction traces in Appendix G. Appendices A and B document benchmark construction, metric definitions, evaluation protocols, and evaluator prompts. Appendices C and D detail cost accounting, failure handling, and the human post-editing study. Upon acceptance, we will release the FigStruct implementation, the complete 150-pair benchmark, model outputs eligible for redistribution, and run configurations specifying software dependencies, model versions, and inference parameters.

ETHICS STATEMENT

Researchers should check generated figures for textual, factual, and logical errors before scientific use. Participants gave informed consent after receiving the research purpose and task requirements and could withdraw from the study. Redrawing and post-editing participants were compensated at prevailing market rates. Names and contact details were deleted after the study, and participant records were indexed by anonymous identifiers. As agreed with participants, we report only aggregate participant statistics and human-study results and will not release individual questionnaire responses or personal information. Redrawing participants consented to the public release of their de-identified drawings. All 150 reference figures are licensed under CC BY 4.0. Released figures and redrawings will include source attribution, license information, and modification descriptions.

REFERENCES

Maneesh Agrawala, Wilmot Li, and Floraine Berthouzoz. Design principles for visual communication. *Commun. ACM*, 54(4):60–69, 2011.

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*, 2024a. URL https://openreview.net/forum?id=v3K5TVP8kZ.

Jonas Belouadi, Simone Paolo Ponzetto, and Steffen Eger. DeTikZify: Synthesizing graphics programs for scientific figures and sketches with TikZ. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b. URL https://openreview.net/forum?id=bcVLFQCOjc.

Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. DeepSVG: A hierarchical generative network for vector graphics animation. In *NeurIPS*, 2020.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14953–14962. IEEE, 2023.

Qijia He, Xunmei Liu, Hammaad Memon, Ziang Li, Zixian Ma, Jaemin Cho, Jason Ren, Daniel S Weld, and Ranjay Krishna. Vfig: Vectorizing complex figures in svg with vision-language models, 2026. URL https://arxiv.org/abs/2603.24575.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *NeurIPS*, 2024.

Siyuan Huang, Yifan Zhou, Yutong Gao, Zi Yin, Juyang Bai, Xinxin Liu, Rama Chellappa, Chun Pong Lau, Cheng Peng, Sayan Nag, and Shraman Pramanick. Scifig: Towards automating editable figure generation for scientific papers, 2026. URL https://arxiv.org/abs/2601.04390.

Ajay Jain, Amber Xie, and Pieter Abbeel. VectorFusion: Text-to-SVG by abstracting pixel-based diffusion models. In *CVPR*, pp. 1911–1920. IEEE, 2023.

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

Tzu-Mao Li, Michal Lukác, Michaël Gharbi, and Jonathan Ragan-Kelley. Differentiable vector graphics rasterization for editing and learning. *ACM Trans. Graph.*, 39(6):193:1–193:15, 2020.

Zhuoling Li, Jiarui Zhang, Ping Hu, Jason Kuen, Jiuxiang Gu, Hossein Rahmani, and Jun Liu. Automatic method illustration generation for ai scientific papers via drawing middleware creation, evolution, and orchestration. *arXiv preprint arXiv:2603.29590*, 2026.

Kevin Qinghong Lin, Yuhao Zheng, Hangyu Ran, Dantong Zhu, Dongxing Mao, Linjie Li, Philip Torr, and Alex Jinpeng Wang. VCode: a multimodal coding benchmark with SVG as symbolic visual representation. *CoRR*, abs/2511.02778, 2025.

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 illustrations via reference-guided styling. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 57–67, 2026.

Yepeng Liu, Dasen Dai, Chengzhi Liu, Yiren Song, Hai Ci, Yu Zhang, Qi Zhang, Mike Zheng Shou, Xin Eric Wang, and Yuheng Bu. Figures as programs: Recursive generation of editable scientific figures, 2026. URL https://arxiv.org/abs/2609.01006.

Xu Ma, Yuqian Zhou, Xingqian Xu, Bin Sun, Valerii Filev, Nikita Orlov, Yun Fu, and Humphrey Shi. Towards layer-wise image vectorization. In *CVPR*, pp. 16293–16302. IEEE, 2022.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Ishani Mondal, Zongxia Li, Yufang Hou, Anandhavelu Natarajan, Aparna Garimella, and Jordan L. Boyd-Graber. SciDoc2Diagrammer-MAF: Towards generation of scientific diagrams from documents guided by multi-aspect feedback refinement. In *EMNLP (Findings)*, volume EMNLP 2024 of *Findings of ACL*, pp. 13342–13375. Association for Computational Linguistics, 2024.

Helen C. Purchase. Which aesthetic has the greatest effect on human understanding? In *GD*, volume 1353 of *Lecture Notes in Computer Science*, pp. 248–261. Springer, 1997.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16175–16186. IEEE, 2025.

Nicolas P Rougier, Michael Droettboom, and Philip E Bourne. Ten simple rules for better figures. *PLoS computational biology*, 10(9):e1003833, 2014.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Yasheng Sun, Zezi Zeng, Yifan Yang, Chong Luo, Wenyi Wang, Ziwei Liu, and Jürgen Schmidhuber. Scidiagramedit: Learning to edit scientific diagrams from paper revisions, 2026. URL https://arxiv.org/abs/2607.15272.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via Python execution for reasoning. In *ICCV*, pp. 11854–11864. IEEE, 2023.

Cheng Tan, Qi Chen, Jingxuan Wei, Gaowei Wu, Zhangyang Gao, Siyuan Li, Bihui Yu, Ruifeng Guo, and Stan Z. Li. SketchAgent: Generating structured diagrams from hand-drawn sketches. In *IJCAI*, pp. 1918–1926. ijcai.org, 2025.

VisionCortex. VTracer: Raster to vector graphics converter, n.d. URL https://github.com/visioncortex/vtracer. Software repository, accessed September 15, 2026.

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 *CVPR*, pp. 13315–13325. Computer Vision Foundation / IEEE, 2025.

Ronghuan Wu, Wanchao Su, and Jing Liao. Chat2svg: Vector graphics generation with large language models and image diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23690–23700. IEEE, 2025.

Xueqing Wu, Ashwin Balasubramanian, Bingxuan Li, Dawei Zhu, Kai-Wei Chang, Yale Song, Yiwen Song, Rui Meng, Tomas Pfister, and Nanyun Peng. Paperbanana-interact: Scientific diagram refinement with multi-turn human feedback, 2026. URL https://arxiv.org/abs/2608.30241.

Ximing Xing, Haitao Zhou, Chuang Wang, Jing Zhang, Dong Xu, and Qian Yu. SVGDreamer: Text guided SVG generation with diffusion model. In *CVPR*, pp. 4546–4555. IEEE, 2024.

Ximing Xing, Juncheng Hu, Guotao Liang, Jing Zhang, Dong Xu, and Qian Yu. Empowering LLMs to understand and generate complex vector graphics. In *CVPR*, pp. 19487–19497. Computer Vision Foundation / IEEE, 2025.

Yiying Yang, Wei Cheng, Sijin Chen, Xianfang Zeng, Fukun Yin, Jiaxu Zhang, Liao Wang, Gang Yu, Xingjun Ma, and Yu-Gang Jiang. Omnisvg: A unified scalable vector graphics generation model. *Advances in Neural Information Processing Systems*, 38:113670–113696, 2026.

Katherine Ye, Wode Ni, Max Krieger, Dor Ma'ayan, Jenna Wise, Jonathan Aldrich, Joshua Sunshine, and Keenan Crane. Penrose: from mathematical notation to beautiful diagrams. *ACM Trans. Graph.*, 39(4):144, 2020.

Leixin Zhang, Steffen Eger, Yinjie Cheng, WEIHE ZHAI, Jonas Belouadi, Fahimeh Moafian, and Zhixue Zhao. Scimage: How good are multimodal large language models at scientific text-to-image generation? In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ugyqNEOjoU.

Zongyun Zhang, Jiacheng Ruan, Xian Gao, Ruizhu Zhou, Lingcheng Meng, Lining Hu, Ting Liu, and Yuzhuo Fu. Edit2tikz: A comprehensive and challenging benchmark for scientific figure editing with tikz. *arXiv preprint arXiv:2608.13441*, 2026.

Haozhe Zhao, Shuzheng Si, Zhenhailong Wang, Zheng Wang, Liang Chen, Xiaotong Li, Zhixiang Liang, Maosong Sun, and Minjia Zhang. Crafter: A multi-agent harness for editable scientific figure generation from diverse inputs, 2026. URL https://arxiv.org/abs/2605.30611.

Dawei Zhu, Rui Meng, Yale Song, Xiyu Wei, Sujian Li, Tomas Pfister, and Jinsung Yoon. Paperbanana: Automating academic illustration for AI scientists. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=FneePKFVHT.

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*, 2026b. URL https://openreview.net/forum?id=5N3z9JQJKq.

A Dataset Construction

We construct a dataset of 150 paired scientific figures, each comprising a published reference figure and a manually redrawn input figure. Redrawing simplifies visual presentation while preserving the text and logical structure of each reference figure. All 150 pairs are used for evaluation.

Sources and scope. We collected figures from articles in *Artificial Intelligence Review* (AIR) through the journal website.[1] Collection covered the five-year publication window from March 2021 to March 2026, traversing articles in reverse chronological order. Figures were downloaded in batches directly from article webpages, without extracting them from PDFs. We reviewed figures across the full time window, selecting information-dense scientific diagrams with complex logical structures and visually appealing color schemes. We also considered suitability for SVG reconstruction. We excluded visually simple figures, standalone statistical charts or experimental data plots such as line charts, and other figures unsuitable for reconstruction. Composite figures combining charts with structured diagrams remained eligible for inclusion, as coordinating their heterogeneous visual components introduces additional reconstruction challenges.

Reference-figure selection. Candidate figures were assessed by three raters recruited in the same recruitment campaign as the 50 redrawing participants, with no overlap between the two groups. All raters were required to be proficient in using PowerPoint for scientific figure preparation and able to understand the basic content of complex English-language scientific diagrams. Each candidate was evaluated by all three raters, who assigned a single overall score on a five-point rating scale considering both visual appeal and information density. Raters assessed the figures independently without seeing scores from other raters. We averaged the three ratings for each candidate and ranked the candidates by their mean scores. An author selected the top 150 as reference figures.

Participant recruitment and assignment. Both groups were recruited through campus questionnaires and online part-time job postings using the same eligibility criteria. Applicants were screened for their ability to understand the general meaning of a complex English-language scientific diagram. No particular educational or professional background was required. The author responsible for reference-figure selection assigned three reference figures to each of the 50 redrawing participants. These participants received written instructions to redraw the figures in Microsoft PowerPoint as simplified diagrams that retained their information, and submitted the results as PNG images.

Visual simplification rules. Redrawing participants could change the layout, shapes, font sizes, and spacing, provided that all textual information and logical relationships were preserved. Where color did not carry information, redrawings used black outlines and text without additional colors. Where color was necessary to preserve information, participants used a fixed seven-color rainbow palette. Alternative palettes and transparency adjustments were not permitted. Nonessential icons were removed. Icons needed to convey meaning were replaced by minimal symbols or basic geometric shapes, such as triangles or rectangles, with explanatory text beneath them. These instructions produced input figures with simplified visual styling while retaining the content to be communicated.

Quality control and compensation. The author responsible for reference-figure selection reviewed every redrawing against its reference, checking visual simplicity, color use, icon treatment, logical structure, and textual information. Redrawings that did not meet these requirements were returned for revision and checked again until accepted. Each redrawing participant completed three accepted figures, yielding 150 pairs in total. Redrawing participants were compensated at prevailing market rates for figure drawing on a per-figure basis, with payment issued for each figure after it met the acceptance requirements.

Excluding source papers from web retrieval. Our agent analyzes each input figure to identify presentation problems and then searches the web for relevant design references, downloading candidate papers as PDFs. We recorded the source-paper titles associated with all 150 reference figures. After candidate PDFs were downloaded, papers whose titles appeared in this list were removed from the retrieved reference pool. Paper-level exclusion used the source-title list covering all 150 reference figures.

[1] `https://link.springer.com/journal/10462`

B Evaluation Metrics and Protocols

We evaluate visual quality, information fidelity, and SVG editability on the 150-figure benchmark. This section specifies the reference inputs, annotation procedures, metric definitions, and aggregation rules. Evaluator instructions appear in Section B.5.

B.1 Evaluation inputs and annotation protocol

Evaluation coverage. All methods are evaluated on the same 150 benchmark samples, using the stage-specific inputs defined below. Failed attempts are retried until a final output is obtained for each sample. For quality evaluation, let $\mathcal{I} = \{1, \ldots, N\}$ index the shared benchmark samples, where $N = 150$. Each method contributes a final output for every sample in this set. Quality metrics are computed from the final outputs for all 150 samples. Per-image quality scores are averaged over this full set, while annotation-based metrics pool counts across the same set, subject to metric-specific applicability rules. Counts of inputs affected by generation errors are reported separately.

Stage-specific references. Let X_i denote the manually redrawn input figure for sample i, B_{im} the raster image produced by image-generation method m, and S_{im} the final SVG produced by method m. The first visual-quality evaluation compares direct image generation with generation guided by agent-based image understanding, web retrieval, and prompt construction, using X_i and B_{im} as inputs. The second visual-quality evaluation assesses the rendered final SVG against the same input figure X_i, covering complete agent systems, controller substitutions, and direct generation of a beautified, editable SVG. Both evaluations use the input figure to establish the design starting point and information-density context.

For information fidelity, the reference depends on the evaluation setting. All methods in the composite-quality comparison in Figure 4, as well as direct SVG generation, FigStruct controller comparisons, and component ablations, are evaluated against the original input figure X_i. In the SVG reconstruction comparison in Table 2, FigStruct, AutoFigure, Crafter, VFIG-4B, and VTracer use the same design image from Step 1.C as their reconstruction input and evaluation reference. We denote this shared reconstruction reference by R_i. Rendered similarity and text preservation in this comparison are evaluated against R_i.

Model-based assessment. All visual scoring and information annotation use `gpt-5.6-sol` with `high` reasoning effort. Each visual input contains one reference–candidate pair, with the reference on the left and the candidate on the right. Both images retain their native pixel dimensions and aspect ratios. The pair is encoded as WebP at quality 75 without resizing the combined image. Evaluators are not supplied with candidate method names, generating-model identities, source paths, or outputs from other evaluations. The reference and candidate roles remain visible. For each aesthetic evaluation, every reference and candidate pair is assessed independently three times using the same model and prompt. The reported scores average these three assessments.

Annotation and human verification. For information-based metrics, the model identifies reference text, semantic units, and relations, and records their candidate correspondences, omissions, and errors. These annotations are manually reviewed for missing or incorrect reference items and for incorrect candidate matches or error judgments. Metric values are then computed programmatically from the reviewed annotations. For metrics requiring native SVG representation, the rendered candidate is also associated with SVG DOM evidence. The model produces item-level annotations from which Unit, Relation, and Order are computed.

B.2 Visual-quality evaluation

B.2.1 Five-dimensional evaluation of generated raster images

We assess *Clarity*, *Logical Flow* (Log. Flow), *Aesthetic Quality* (Aes. Qual.), *Visual Expressiveness* (Vis. Expr.), and *Professional Polish* (Polish). Each independent evaluation assigns a score in $[0, 10]$, with one decimal place, to both the input figure and the generated image along all five dimensions. Table 5 defines the dimensions for both visual-quality evaluations.

Table 5: Dimensions used in the two visual-quality evaluations. Raster-image scores use a 0–10 scale; final-output scores use a 0–100 scale.

Dimension	Assessment criterion
Generated raster images: five dimensions	
Clarity	Legibility, distinguishable modules, local comprehension, and freedom from crowding or occlusion.
Logical Flow	Reading order, grouping, hierarchy, and spatial organization that supports the scientific logic.
Aesthetic Quality	Color harmony, balanced composition, alignment, spacing, whitespace, and visual appeal.
Visual Expressiveness	Meaningful visual encoding that makes abstract content more intuitive.
Professional Polish	Consistency and refinement of typography, lines, arrows, shapes, and local details.
Final SVG renders: seven dimensions	
Layout	Composition, balance, space utilization, and placement of the main content.
Spacing	Alignment, margins, inter-element spacing, and consistency of spatial distribution.
Hierarchy	Clear differentiation of titles, primary content, and secondary information.
Color	Harmony, contrast, semantic function, and consistent use of color.
Typography	Consistency of font families, sizes, weights, and line spacing.
Readability	Ease of reading text, distinguishing nodes and arrows, and following visual flow.
Professionalism	Visual completeness, coherence, refinement, and publication readiness.

Shared input-figure baseline. Let a_{imdr} and b_{imdr} denote the input-figure and generated-image scores for sample i, method m, dimension d, and evaluation $r \in \{1, 2, 3\}$. For the set of compared methods $\mathcal{M}$, we pool all repeated ratings of the same input figure to obtain a common baseline:

$$\bar{a}_{id} = \frac{1}{3|\mathcal{M}|} \sum_{m \in \mathcal{M}} \sum_{r=1}^{3} a_{imdr}, \qquad \bar{b}_{imd} = \frac{1}{3} \sum_{r=1}^{3} b_{imdr}. \tag{2}$$

All methods therefore use the same baseline for a given sample and dimension. The reported generated-image score and absolute gain are

$$\mathrm{Score}_{md} = \frac{1}{N} \sum_{i=1}^{N} \bar{b}_{imd}, \qquad \mathrm{Gain}_{md} = \frac{1}{N} \sum_{i=1}^{N} (\bar{b}_{imd} - \bar{a}_{id}), \tag{3}$$

where $N = 150$. Visual-quality gains are reported as absolute score differences on the ten-point scale. The five dimensions are reported separately. Within-evaluation differences elicited by the prompt serve the paired assessment. Reported gains use the shared baseline in Equation 2.

B.2.2 Seven-dimensional evaluation of final SVG renders

The final-output evaluation uses seven dimensions: Layout, Spacing, Hierarchy, Color, Typography, Readability, and Professionalism. Each of the three independent evaluations assigns an integer score in $[0, 100]$ to the candidate along each dimension. The input figure provides context, while the reported value measures the absolute visual quality of the candidate. In particular, this score is not a difference between the candidate and the input figure. For scores s_{imdr}, the dataset-level aesthetic score is

$$\mathrm{Aesthetic}_m = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{7} \sum_{d=1}^{7} \frac{1}{3} \sum_{r=1}^{3} s_{imdr} \right). \tag{4}$$

Thus, Aesthetic Quality in the five-dimensional evaluation is a single 0–10 dimension, whereas the final-output Aes. Qual. score averages seven 0–100 dimensions. The latter evaluates visual defects such as clutter, weak contrast, and poor alignment. Fidelity metrics assess semantic errors separately.

B.3 INFORMATION FIDELITY FROM REVIEWED ANNOTATIONS

In this and the following subsection, metrics are computed separately for each method. We omit the method index on metric values and annotation counts for brevity. Unless a metric specifies an applicable subset, sums over i cover the full benchmark set $\mathcal{I}$ defined in Section B.1.

B.3.1 TEXT, UNITS, RELATIONS, AND ORDER

Text occurrences and matching. A text occurrence is a visible text block with a distinct semantic role. Line wrapping within a block is merged, while repeated strings in different nodes or roles remain separate occurrences. Reference–candidate matches are one-to-one and require correct semantic ownership. A candidate occurrence cannot satisfy multiple reference occurrences. Reference text that remains unreadable is marked ambiguous and excluded on both sides. Required reference occurrences contribute to recall. Legitimate non-required occurrences may contribute to precision without creating a recall requirement.

Text credit requires both a readable, semantically correct rendering and a binding to visible native SVG `<text>` elements, including their `<tspan>` content. Text present only inside raster assets or as path outlines does not receive native text credit and is excluded from the native candidate-text denominator. Native text occurrences in the final SVG that are unsupported by the reference or duplicated without justification remain in the candidate-text count but receive zero matching credit. Repositioning or line wrapping alone does not reduce credit.

Text Precision, Recall, and F1. For sample i, C_i^t counts eligible native candidate occurrences, and G_i^t counts required reference occurrences. Let $S_i^{t,P}$ denote matched credit summed over all legitimate reference text, and $S_i^{t,R}$ the sum restricted to required reference text. Fully correct, readable matches receive credit 1. Readability defects use the anchors 0.98, 0.95, 0.90, 0.85, 0.70, 0.50, 0.25, and 0. Character-level differences may receive intermediate credit according to retained information. We first pool annotation counts and credit across all 150 samples, then compute

$$P_t = \frac{\sum_i S_i^{t,P}}{\sum_i C_i^t}, \qquad R_t = \frac{\sum_i S_i^{t,R}}{\sum_i G_i^t}, \qquad F_t = \frac{2P_tR_t}{P_t + R_t}. \tag{5}$$

This is a micro-aggregated evaluation. Let T denote the equal-weight mean of the three quantities. The displayed *Text Accuracy* expresses T on a 0–100 scale:

$$T = \frac{P_t + R_t + F_t}{3}, \qquad \mathrm{TextAccuracy} = 100T. \tag{6}$$

It is a composite text-preservation score, rather than a character-level exact-match rate.

Unit. Semantic units comprise meaningful modules, nodes, labels, formulas, or other independently interpretable information units. Purely decorative objects are excluded. Each one-to-one matched unit receives credit 1 for complete preservation, 0.5 for partial preservation, and 0 for an incorrect or missing unit. Let S_i^u be the sum of unit credit, and C_i^u and G_i^u the candidate and required-reference unit counts. The reported Unit score is $100F_u$, where

$$P_u = \frac{\sum_i S_i^u}{\sum_i C_i^u}, \qquad R_u = \frac{\sum_i S_i^u}{\sum_i G_i^u}, \qquad F_u = \frac{2P_uR_u}{P_u + R_u}. \tag{7}$$

Visual fragmentation of one concept does not create multiple semantic units. Distinct concepts within one container may be matched separately.

Relation. A relation is correct only when its source unit, target unit, direction, and relation type are all correct. Directed connections, grouping, and containment are included where required by the reference. Wrong endpoints, reversed directions, and incorrect relation types receive no partial credit. Let K_i^r be the number of correct relations and C_i^r and G_i^r the candidate and reference relation counts. The reported Relation score is $100F_r$, with

$$P_r = \frac{\sum_i K_i^r}{\sum_i C_i^r}, \qquad R_r = \frac{\sum_i K_i^r}{\sum_i G_i^r}, \qquad F_r = \frac{2P_rR_r}{P_r + R_r}. \tag{8}$$

Alternative layouts and connector routes are allowed when the semantic relation remains correct.

Order. Reference annotations specify meaningful precedence, stage, numbering, or hierarchical-order constraints. Let K_i^o and G_i^o denote the satisfied and total reference constraints. Order is computed independently of Relation:

$$O = \frac{\sum_i K_i^o}{\sum_i G_i^o}, \qquad \text{Order} = 100O. \tag{9}$$

A constraint whose participating unit is missing is unsatisfied. Figures without a required order contribute no order constraints.

Fidelity aggregate. After pooling the reviewed annotations, we combine the dataset-level text composite T, Unit F1, Relation F1, and Order score:

$$\text{Fidelity} = 100\,(0.35T + 0.25F_u + 0.30F_r + 0.10O)\,. \tag{10}$$

The text component T is the equal-weight mean of Text Precision, Recall, and F1, corresponding to Text Accuracy in Equation 6 normalized to $[0, 1]$. All four components are on a $[0, 1]$ scale before the final multiplication by 100. Missing matches in an applicable component receive zero credit. An F1 score with zero precision and recall is zero. For an applicable text, unit, or relation component with reference targets but no eligible candidate items, precision and F1 are set to zero. If a component has no eligible reference targets across the evaluated dataset, it is not applicable, and the remaining Fidelity weights are renormalized in proportion to their original values.

B.3.2 Reconstruction similarity and text preservation

Rendering protocol. In the SVG reconstruction comparison, image similarity is measured between the shared Step 1.C design image R_i and a standardized render of the final SVG from each method. The SVG is rendered with CairoSVG, preserving its viewBox aspect ratio, scaled to fit the reference canvas, and centered on a white background. The full canvas is evaluated without masking internal regions or applying image registration. For SSIM and LPIPS, both images are downsampled by a factor of two in each dimension using Lanczos resampling.

SSIM. RGB intensities are scaled to $[0, 1]$. Local means, variances, and covariance are computed with an 11×11 Gaussian window of standard deviation 1.5. For local image signals x, y,

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{11}$$

where $C_1 = 0.01^2$ and $C_2 = 0.03^2$. The per-image score averages the local maps over RGB channels and the full image, with higher values indicating greater similarity.

DINO. We use DINOv2-base (`facebook/dinov2-base`) patch features. Each full image is resized with bicubic interpolation, preserving its aspect ratio, and padded with white to 518×518 pixels. Inputs use ImageNet channel means $(0.485, 0.456, 0.406)$ and standard deviations $(0.229, 0.224, 0.225)$. We remove the final-layer CLS token and ℓ_2-normalize the 37×37 patch features. For corresponding reference and candidate patch features f_p^R, f_p^S,

$$\text{DINO}_i = \frac{\sum_p w_p \langle f_p^R, f_p^S \rangle}{\sum_p w_p}, \tag{12}$$

where w_p is the fraction of the 14×14 patch covered by the reference image rather than external padding. Higher similarity is better.

LPIPS. We use the AlexNet-based LPIPS model, version 0.1, through `lpips` 0.1.4 with spatial output enabled. RGB inputs are mapped to $[-1, 1]$. The spatial distance map is bilinearly resized to the evaluation image size when necessary, and its mean over the full canvas gives the per-image distance. Lower is better. SSIM, DINO, and LPIPS are each averaged over all 150 samples after their per-image values are computed.

Semantic text preservation (Sem.). For the SVG reconstruction comparison, the shared Step 1.C design image R_i is decomposed into independently meaningful text or symbol items. Line-wrapped phrases are merged, while repeated labels at different semantic positions are counted separately. An item passes when its meaning is correctly retained, including unambiguous synonyms or minor spelling differences that preserve meaning. Missing, semantically incorrect, or unreadable items fail. With K_i^s correct items among G_i^s eligible reference items,

$$\text{Sem.} = 100\,\frac{\sum_i K_i^s}{\sum_i G_i^s}. \tag{13}$$

Text-layout correctness (Layout). Using the same reference-item denominator, let K_i^ℓ count items assigned to the correct semantic module, node, or process role in the candidate. Then

$$\text{Layout} = 100\,\frac{\sum_i K_i^\ell}{\sum_i G_i^s}. \tag{14}$$

This measures semantic placement rather than bounding-box overlap or the aesthetic Layout dimension. A valid rearrangement may pass, whereas assignment to the wrong module fails. Both Sem. and Layout use model-produced, manually reviewed item annotations.

B.4 SVG EDITABILITY AND REPRESENTATION DIAGNOSTICS

B.4.1 NATIVE REPRESENTATION AND EDITABILITY

Text Edit. Let V_i be the number of all visible semantic text or symbol items in the final output and V_i^{native} the number represented by native `<text>`/`<tspan>` elements. The strict native-text editability rate is

$$\text{TextEdit.} = 100\,\frac{\sum_i V_i^{\text{native}}}{\sum_i V_i}. \tag{15}$$

Text inside raster assets, path outlines, or symbols assembled from non-text shapes is not native editable text. The denominator concerns the output: missing reference text does not enter it, while additional visible candidate text does. This distinguishes Text Edit. from reference-based text recall.

SVG Editability. For each image, we calculate three components in $[0, 1]$. *Editable Text* E_i is the matched native credit for required reference text divided by the number of required eligible reference occurrences. *Native Text Exact* X_i^e is the fraction of these occurrences that have an exact native-text match with correct semantic ownership and visible rendering. Exact matching permits only Unicode NFKC normalization, whitespace collapse, and trimming. It does not ignore case, punctuation, numbers, or operators.

Object Separability Q_i measures whether required semantic units and relations can be edited independently. Each target is bound to visible SVG elements or groups using anonymized Chromium DOM evidence. Three independent evaluations assign target credit 1 for independent editability, 0.5 for editable but insufficiently grouped or partly shared elements, and 0 for a missing, raster-only, inseparable, or ungrounded target. Let N_i^{obj} be the number of required targets and q_{itr} the credit for target t in evaluation $r \in \{1, 2, 3\}$ of sample i. Then

$$Q_i = \frac{1}{N_i^{\text{obj}}} \sum_{t=1}^{N_i^{\text{obj}}} \frac{1}{3} \sum_{r=1}^{3} q_{itr}, \qquad \text{SVGEditability}_i = 100(0.25E_i + 0.25X_i^e + 0.50Q_i). \tag{16}$$

DOM checks exclude hidden elements, unused definitions, empty groups, and elements outside the canvas. If the reference image contains no eligible targets for a component, that component is not applicable. The remaining weights are rescaled proportionally to sum to one. Reference targets omitted from the candidate still receive zero credit. The reported SVG Editability is the mean of per-image aggregate scores over applicable samples within the full 150-input benchmark. Its per-image Editable Text component uses local reference counts. It differs from both the output-based Text Edit. rate and the dataset-level micro Text Recall.

Clean. For sample i, let N_i^{basic} count `rect`, `circle`, and `ellipse` elements; N_i^{line} count `line` and `polyline`; and N_i^{path} count `path` and `polygon`. The per-image structural diagnostic is

$$\text{Clean}_i = \frac{N_i^{\text{basic}} + N_i^{\text{line}}}{N_i^{\text{basic}} + N_i^{\text{line}} + N_i^{\text{path}} + 10^{-8}}. \tag{17}$$

Counts are obtained by XML parsing. Text and image elements do not enter this formula. A high value indicates a larger fraction of basic geometric elements and connectors, not semantic correctness.

Raster area. For root canvas $\mathcal{C}_i$ and transformed declared image rectangles $\mathcal{R}_{ij}$, where j indexes image elements,

$$\text{RasterArea}_i = 100\,\frac{\text{area}(\mathcal{C}_i \cap \bigcup_j \mathcal{R}_{ij})}{\text{area}(\mathcal{C}_i)}. \tag{18}$$

The implementation accumulates ancestor and element transforms and approximates the union on a 512×512 grid. Transparent pixels inside a declared image rectangle are included. Raster area characterizes raster coverage rather than reference-asset retention or an error rate.

Path Cmds. We count SVG path command letters `MmLlHhVvCcSsQqTtAaZz` in the `d` attributes of all `path` elements and sum these counts for each image. Implicit coordinate repetitions and polygon or polyline points do not add command letters. This is an encoding-dependent complexity diagnostic. Clean, Raster area, and Path Cmds are reported as means of per-image values over all 150 samples.

B.4.2 Rendered SVG geometry

Geometry is extracted from the final SVG using `getBBox()` in Chromium and transformed into root viewBox coordinates. Hidden elements and unused definitions are excluded. For global geometry, completely off-canvas image sentinels occupying at most 0.05% of the canvas area are excluded. Let $\mathcal{B}_i$ be the smallest axis-aligned bounding rectangle enclosing the remaining visible graphics, and $\mathcal{C}_i$ the canvas. We use $\mathbf{1}[\cdot]$ for the indicator function, which equals 1 when the enclosed condition holds and 0 otherwise.

GFR and OAR. Per-image global fit and out-of-canvas area are

$$\text{GFR}_i = \mathbf{1}[\mathcal{B}_i \subseteq \mathcal{C}_i], \qquad \text{OAR}_i = \frac{\text{area}(\mathcal{B}_i) - \text{area}(\mathcal{B}_i \cap \mathcal{C}_i)}{\text{area}(\mathcal{B}_i)}. \tag{19}$$

GFR is higher-is-better and OAR is lower-is-better. OAR uses the enclosing content rectangle, rather than the sum of individual overflow areas.

Automatic text–container assignment. Each native-text bounding box is assigned to the smallest rectangular container that contains its center and occupies at most 5% of the canvas. Container interiors account for half the boundary stroke width. Text boxes assigned to the same container are merged before evaluating containment and padding. These assignments are automatic geometric proxies. Let N_i^{grp} be the number of assigned groups, $\mathcal{T}_{ig}$ their merged text boxes, $\mathcal{K}_{ig}$ their container interiors, and p_{ig} their minimum signed padding. Here $g \in \{1, \ldots, N_i^{\text{grp}}\}$ indexes groups. The quantity p_{ig} is the minimum of the left, right, top, and bottom clearances between the merged text box and its container interior. A negative value indicates overflow.

TBR. The containment rate is

$$\text{TBR}_i = \frac{1}{N_i^{\text{grp}}} \sum_{g=1}^{N_i^{\text{grp}}} \mathbf{1}[\mathcal{T}_{ig} \subseteq \mathcal{K}_{ig}]. \tag{20}$$

TPVR-6px and TPVR-0.20em. Distances are scaled by $s_i = \min(1344/W_i, 768/H_i)$ for canvas width W_i and height H_i. Writing $\tilde{p}_{ig} = s_i p_{ig}$ and $\tilde{f}_{ig}$ for the scaled maximum font size within a group,

$$\text{TPVR-6px}_i = \frac{1}{N_i^{\text{grp}}} \sum_{g=1}^{N_i^{\text{grp}}} \mathbf{1}[\tilde{p}_{ig} < 6], \tag{21}$$

$$\text{TPVR-0.20em}_i = \frac{1}{N_i^{\text{grp}}} \sum_{g=1}^{N_i^{\text{grp}}} \mathbf{1}[\tilde{p}_{ig} < 0.20\tilde{f}_{ig}]. \tag{22}$$

The former uses a fixed normalized pixel margin, and the latter uses a font-relative margin. TBR and both TPVR variants share the same applicable groups and samples.

Overlap. Overlap uses individual native `<text>` element boxes before text–container grouping, denoted by $\mathcal{U}_{ia}$ for element a in sample i. For two distinct boxes $\mathcal{U}_{ia}, \mathcal{U}_{ib}$ with positive areas and $a < b$, an overlap event occurs when

$$\frac{\text{area}(\mathcal{U}_{ia} \cap \mathcal{U}_{ib})}{\min(\text{area}(\mathcal{U}_{ia}), \text{area}(\mathcal{U}_{ib}))} \geq 0.05. \tag{23}$$

The per-image Overlap score is the fraction of native-text boxes involved in at least one such event. Each affected box is counted once. Geometry metrics are averaged over applicable samples within the full 150-input benchmark. TBR and TPVR are not applicable without eligible groups, and Overlap is not applicable without native-text boxes. In particular, path-outlined text does not establish a zero violation rate.

B.5 Evaluator instructions

The following English instructions consolidate the criteria and annotation requirements for the reported metrics. Execution paths and unused diagnostic outputs are omitted. Numerical aggregation follows the equations above and is performed by the evaluation scripts rather than by the evaluator.

B.5.1 Five-dimensional raster-image evaluation

You are a strict reviewer of scientific diagrams. You receive two images: **A**, a rough diagram expressing the author's core content, main modules, and basic logic; and **B**, an anonymous regenerated diagram. The rough diagram establishes the intended content but may be an incomplete or imperfect presentation of it. The goal is to communicate the scientific content more clearly and professionally.

General principles. Substantial changes to layout, module placement, color, visual form, and information organization are permitted when the main scientific meaning and relationships remain correct. Do not penalize a candidate merely for looking different from A. Scientifically plausible additions consistent with the diagram's topic may improve its explanatory quality; uncertain additions receive no extra credit merely for being present. Judge meaningful communication and visual quality, and identify observable problems in your evidence. Do not infer the generating method or model identity.

For each of the following five dimensions, independently score *both A and B* on a 0.0–10.0 scale, allowing one decimal place.

Clarity. Assess readable text, distinguishable figures and modules, crowding, occlusion, local legibility, and how quickly each region can be understood. Anchors: 10, exceptionally clear with almost no reading difficulty; 8, clear overall with minor local problems; 6, understandable but with evident crowding or blur; 4, multiple difficult regions; 2, mostly unclear; 0, unreadable.

Logical Flow. Assess reading order, module grouping, parent–child hierarchy, traceable information flow, and whether spatial organization supports the scientific logic. Distinguish a change of visual style from a change in logical organization. For the within-evaluation difference between B and A, changes only to colors, fonts, borders, shadows, or line styles should normally yield at most 1.5 points of improvement. Minor spacing adjustments or local movements should normally yield at most 3.0 points. An improvement of 4.0 or more requires substantial, effective reorganization of modules, hierarchy, grouping, information flow, or overall composition. Large changes that make the diagram more confusing should not receive a high score.

Aesthetic Quality. Assess color harmony, compositional balance, alignment, spacing, whitespace, typography and shape style, and overall visual appeal. A color-only change can meaningfully improve this dimension even without structural reorganization. Anchors: 10, excellent visual quality for a research-paper figure; 8, attractive, harmonious, and professional with minor issues; 6, acceptable but ordinary or inconsistent; 4, visibly rough or discordant; 2, poor visual quality; 0, impossible to assess.

Visual Expressiveness. Assess how well abstract content is expressed through intuitive figures, symbols, hierarchy, contrast, and meaningful visual encodings. Distinguish informative visual redesign from added decoration. For the within-evaluation difference between B and A, changes only to colors, fonts, shadows, or borders should normally yield at most 2.0 points of improvement. Reusing the original module structure with a few decorative additions should normally yield at most 3.0 points. An improvement of 4.0 or more requires a substantially more intuitive, informative, and professional visual representation. More decoration alone does not imply greater expressiveness.

Professional Polish. Assess consistency of fonts, sizes, text styles, lines, arrows, corner radii, borders, alignment, spacing, graphical style, and local finish. Look for generation artifacts and nonstandard or visibly unfinished elements. Anchors: 10, highly refined and nearly ready for publication; 8, professional overall with minor fixes needed; 6, recognizable scientific-diagram form with several rough details; 4, clearly inconsistent or unfinished; 2, very rough; 0, unusable as a scientific figure.

Output. For each dimension, return `original_score`, `generated_score`, the within-evaluation `delta`, and one to three concrete observations supporting the scores. Use the keys `clarity`, `logical_flow`, `aesthetic_quality`, `visual_expressiveness`, and `professional_polish`. Do not combine the five dimensions into an overall score.

The within-evaluation delta restrictions guide each reviewer when comparing A and B. Reported gains are recomputed from the three candidate ratings and the shared input-figure baseline, as specified in Section B.2.1.

B.5.2 SEVEN-DIMENSIONAL FINAL-OUTPUT EVALUATION

You are one of three independent reviewers assessing scientific-diagram presentation quality. The left image is the rough reference and the right image is an anonymous final candidate. You are not given candidate method identities, model identities, prior scores, or the other reviewers' judgments. Use the supplied native-resolution WebP pair as the visual evidence.

Assign seven independent *integer* scores from 0 to 100 to the candidate: **Layout**, composition, balance, space use, and placement; **Spacing**, alignment, margins, and distribution consistency; **Hierarchy**, clear separation of primary and secondary information; **Color**, harmony, contrast, function, and consistency; **Typography**, font, size, weight, and line-spacing consistency; **Readability**, ease of reading text, nodes, arrows, and visual flow; and **Professionalism**, visual completeness, coherence, refinement, and publication quality.

Score the candidate's *absolute visual quality*. The rough reference supplies the design starting point and information-density context. Do not subtract a reference score or report a percentage improvement.

Use these quality bands: 98–100, virtually no observable defects; 95–97, excellent with nearly imperceptible blemishes; 90–94, clear and professional with a few minor issues; 85–89, professional overall with localized problems; 80–84, good with several observable flaws; 75–79, usable but insufficiently refined; 70–74, fairly clear with evident roughness; 60–69, basically usable but ordinary or inconsistent; 50–59, barely usable with crowding, imbalance, or legibility problems; 40–49, several regions substantially impede understanding; 20–39, severely confused and only partly usable; 1–19, broad visual failure; 0, impossible to evaluate.

Boundary with fidelity. Semantic mistakes in words, numbers, formulas, node meanings, or connections are handled by the fidelity evaluation and must not mechanically determine aesthetic scores. Assess their observable visual consequences where relevant: overlap, tiny text, weak contrast, crossings, clutter, broken alignment, abnormal whitespace, or disrupted hierarchy. Deleting information must not earn an artificial cleanliness bonus. One defect may affect multiple dimensions only when you explain its distinct visual consequence for each.

Output. Return the seven scores under `layout`, `spacing`, `hierarchy`, `color`, `typography`, `readability`, and `professionalism`. Provide a short, observable, location-specific reason for each score. Choose the relevant quality band before selecting an integer within it. Do not mechanically assign the same score to all dimensions, infer the generating method, or rank the candidate against unseen outputs.

B.5.3 INFORMATION-FIDELITY ANNOTATION

You are annotating information preservation between a reference diagram on the left and an anonymous candidate render on the right. Use the supplied pair and candidate DOM evidence. Do not infer method identities or use another source image to correct the supplied reference. Record explicit items and matches; do not assign holistic Unit, Relation, or Order scores.

Reference inventory. Enumerate readable semantic text occurrences, meaningful units, required relations, and explicit ordering constraints. Assign stable IDs, distinguish required from non-required text, and flag items that cannot be reliably interpreted. Merge line wrapping within one text block; count repeated labels in different semantic positions separately. Ignore pure decoration.

Text matching. Match reference occurrences one-to-one by content and semantic ownership. Bind each credited candidate occurrence to visible native SVG text elements using DOM IDs. Adjacent native elements used only for line wrapping may jointly represent one occurrence. Do not reuse a candidate occurrence to satisfy multiple reference items. Raster-internal text and path-outline text may preserve visual information but receive no native-text credit. Count native text in the final SVG that is unsupported by the reference or duplicated without justification as unmatched extra occurrences. Ambiguous reference occurrences are neutral on both sides.

Assign credit according to the actual rendered content and legibility: 1.00, correct and clear; 0.98, nearly imperceptible defect; 0.95, slight defect with effortless reading; 0.90, noticeable small defect with complete content still certain; 0.85, clear overlap or occlusion but still readable; 0.70, severe crowding but decipherable; 0.50, only the gist or part remains readable; 0.25, fragments only; 0.00, missing, wrong, unreadable, or non-native. Character, number, punctuation, and symbol differences may receive intermediate credit supported by the retained information. A correct DOM string does not override an incorrect or unreadable rendering. Record the reference text, candidate text, native element IDs, semantic owner, credit, and reason.

Unit matching. Enumerate the candidate's non-decorative semantic units and match them to reference units. Assign 1 for complete preservation, 0.5 for partial preservation, and 0 for a missing or incorrect unit. List unmatched candidate units so that extra content remains represented in the candidate count. Match by meaning rather than identical position, shape, or number of drawing primitives.

Relation matching. Represent each relation by its endpoints and relation type, including direction when applicable. A relation is correct only if both endpoints, direction, and type agree with the reference. Record missing and additional relations explicitly. Wrong or reversed relations receive no partial credit. Different connector routing or visual layout is acceptable when semantic connections remain correct.

Order constraints. For each explicit reference precedence, stage, numbering, or hierarchical-order requirement, record whether the candidate satisfies it and explain any failure. A constraint fails if a participating unit is missing. Do not derive Order by copying Relation judgments. Where the reference imposes no order, record that the component is not applicable.

Output. Return the reference inventory, candidate inventory, one-to-one matches, native-text bindings, unit credits, relation correctness, satisfied and unsatisfied order constraints, and evidence for omissions or errors. Final metric values will be calculated from the annotations after manual review.

B.5.4 SEMANTIC TEXT, PLACEMENT, AND NATIVE REPRESENTATION

Inspect the shared Step 1.C design image supplied as the reconstruction reference, the anonymous final candidate render, and the candidate SVG DOM. Create an itemized annotation for the following three questions.

Semantic preservation. For every eligible reference text or symbol item, record whether its meaning is correctly retained in the candidate. Unambiguous synonyms and minor spelling differences may preserve meaning. Missing, contradictory, cut-off, or unreadable items fail. Do not substitute character-level exact matching for this semantic judgment.

Semantic placement. For the same reference items, record whether each belongs to the correct module, node, or process role in the candidate. A change of position or scale is acceptable if semantic ownership is preserved. An item assigned to the wrong module fails. Keep content and placement judgments separate when the evidence supports different outcomes.

Native-text representation. Independently enumerate all visible semantic text and symbol items in the candidate, including additional items. For each, determine whether it is represented by native SVG text or tspan content, text inside a raster image, path outlines, or non-text geometric shapes. Only the native-text category counts as editable text. Merge nodes used only to wrap one phrase; do not use raw DOM node counts as semantic item counts.

Output. Return reference-item content and placement labels, the complete visible candidate-text inventory, representation labels, relevant DOM IDs, and evidence. These annotations will be manually checked before the dataset-level counts are pooled.

B.5.5 OBJECT SEPARABILITY

You are one of three independent annotators of SVG object separability. Use the supplied reference–candidate image pair, the required reference-unit and reference-relation inventory, and anonymized DOM evidence. Do not infer the candidate method or inspect another annotator's results.

For every required target, identify the SVG elements or groups that implement it and record whether it can be selected and edited independently. Use credit 1.0 when the complete semantic object can be selected, moved, hidden, or modified without changing other required targets. Use 0.5 when its components are editable but lack suitable grouping or partly share elements with other objects. Use 0.0 for a missing target, a raster-only target, inseparable path geometry, destructive sharing with other targets, or an unreliable target–DOM match.

Assess relation carriers independently, including their connecting paths and arrows. A shared arrowhead-marker definition alone does not invalidate an otherwise independently editable connection. Unused definitions, hidden elements, zero-area objects, empty groups, and decorative wrappers do not establish an editable semantic object. Missing targets must remain in the target list with zero credit.

Output. For every `target_id`, return `candidate_selectors`, `semantic_match`, `independently_selectable`, `shared_with_other_targets`, `credit`, and a concrete `reason`. Assign target-level evidence and credit rather than an overall percentage. The evaluation script checks selector existence, visibility, target coverage, and permitted credit values before aggregation.

C GENERATION COST AND ERROR ANALYSIS

We report API costs for raster-image beautification and SVG generation, and counts of inputs affected by generation errors on the 150-input benchmark.

C.1 COST ACCOUNTING AND GENERATION ERRORS

API cost. Costs are reported in USD and include all billable model calls within the evaluated stage and their retries, but exclude evaluation costs. All methods ultimately produce outputs for the same $N = 150$ benchmark samples. For each sample $i \in \mathcal{I}$, let c_{im} be the total API expenditure incurred by method m within the measured stage until the final output is obtained, including all charges for failed calls and retries. The mean API cost per output is

$$\overline{C}_m = \frac{1}{N} \sum_{i=1}^{N} c_{im}, \qquad N = 150. \tag{24}$$

All 150 inputs contribute to this average, including inputs that encounter generation errors before completing successfully. We use recorded costs or estimates based on token usage and model prices, as specified below. Local inference has no external API charge. Hardware and electricity costs are excluded.

Generation errors. For each input, we record whether generation encounters a model-call error caused by an abnormal response or an invalid escape sequence. Each affected input is counted once, regardless of the number of failed calls or retries for that input. Failed attempts are retried until an output is obtained, so all methods ultimately produce final outputs for all 150 inputs. Network and account-credit errors are excluded from this count. The fraction of affected inputs uses all 150 benchmark inputs as its denominator. Quality metrics are computed from the final outputs for all 150 inputs.

C.2 RASTER-IMAGE BEAUTIFICATION

Table 6 corresponds to the five-dimensional visual-quality comparison. GPT Image 2.0 receives image-and-text input in Direct and text-only input in FigStruct. Image understanding, retrieval, and prompt construction in FigStruct run locally. Both Qwen configurations use one API call costing RMB 0.5, converted at RMB 6.7459 per USD using the September 22, 2026 central parity rate.[2]

Table 6: Mean API cost of raster-image beautification over 150 inputs (USD/image), rounded to four decimal places.

Method	Image generator	API cost
Original	—	—
Direct	Qwen Image 2.0 Pro	0.0741
Direct	GPT Image 2.0	0.0695
PaperBanana	Nano Banana Pro	0.1439†
FigStruct	Qwen Image 2.0 Pro	0.0741
FigStruct	GPT Image 2.0	0.0631

† Mean API cost from runs using the official PaperBanana repository code under our reported configuration, including all billable calls and retries across 150 inputs.

PaperBanana configuration and API cost. Our PaperBanana configuration uses one Gemini 3.8 Flash suggestion call followed by one Nano Banana Pro 1K image-generation call. During preliminary runs with the original PaperBanana suggestion model, Gemini-3-Pro, we observed high response latency and frequent execution failures in our environment. To reduce response latency, we selected Gemini 3.8 Flash, informed by its competitive performance on reasoning and multimodal benchmarks published by Google.[3] We run PaperBanana using its official implementation with the model configuration described above. We sum the charges for all billable API calls incurred during raster-image beautification across the 150 inputs, including failed calls and retries. The reported mean cost per image is this total divided by 150.

[2] State Administration of Foreign Exchange, RMB exchange-rate table, September 22, 2026.

[3] Google DeepMind, Gemini 3.8 Flash model card, accessed September 22, 2026.

C.3 SVG RECONSTRUCTION AND CONTROLLER ABLATIONS

Table 7: Mean SVG-stage API cost, composite quality, and counts of inputs affected by generation errors over all 150 inputs. Composite quality averages fidelity, aesthetic quality, and SVG editability on a 0–100 scale. Total charges for all billable calls, including failed calls and retries, are divided by 150, excluding preceding raster-image generation. Each affected input is counted once regardless of the number of failed calls or retries. All inputs ultimately yield outputs.

Method	USD per output	Composite quality	Affected inputs / 150
AutoFigure	0.2924	82.57	8
Crafter	2.439	79.00	12
Direct (Gemma-4 31B)	0	77.88	14
Direct (GLM-5.3 Flash)	0.003324	81.07	11
Direct (GPT-5.6 Luna)	0.01160	74.83	10
Direct (GPT-5.6 Terra)	0.1160	74.60	9
FigStruct (Gemma-4 31B)	0.009429	85.47	2
FigStruct (GLM-5.3 Flash)	0.2089	87.10	2
FigStruct (GPT-5.6 Terra)	3.696	88.34	1

Baseline model configurations. AutoFigure-Edit, denoted AutoFigure in our comparisons, uses Gemini-3-Pro for SVG reconstruction on all 150 inputs, with API response failures handled through retries. Recurrent response failures in our environment motivated the use of Gemini 3.8 Flash for the suggestion-generation stage of PaperBanana. CraftEditor, denoted Crafter, uses GPT-5.5 to generate and refine SVGs and GPT Image 2.0 to clean images during asset extraction.

Table 7 reports SVG-stage costs, composite quality scores, and counts of inputs affected by generation errors for the baselines and controller ablations. Pipeline costs exclude preceding raster-image generation. Direct methods generate a beautified, editable SVG from the input figure. With Gemma-4 31B, FigStruct incurs API charges only for *Global Refinement* with GPT-5.6 Luna.

Execution reliability and API expenditure. Across 150 inputs, the three FigStruct configurations encounter generation errors on 2, 2, and 1 inputs, compared with 8 for AutoFigure and 12 for Crafter. Each affected input is counted once, and all affected inputs ultimately yield final outputs after retries. API expenditure varies substantially across controllers because iterative review and correction incur repeated calls and cumulative token usage. With the local Gemma controller, these calls run on local hardware, and the paid API expense covers Global Refinement. API-based controllers incur charges throughout the correction rounds, with total expenditure also depending on model-specific pricing. These costs cover external API charges, excluding local inference hardware and electricity.

D HUMAN STUDY OF SVG POST-EDITING EFFORT

We measure the manual editing time required to make generated SVG outputs meet common acceptance criteria in Microsoft PowerPoint. The study covers AutoFigure, Crafter, and the three controller configurations of both Direct and FigStruct reported in Section C. All eight methods are evaluated on the same 50 figures.

D.1 PARTICIPANTS AND TASK ASSIGNMENT

We recruited 400 participants who could understand the basic content of English-language scientific diagrams. None had participated in the dataset-redrawing process described in Section A. We classified participants as frequent or infrequent PowerPoint users through manual review of questionnaire responses. Classification considered experience with scientific figure preparation and publication history, including publication counts. Infrequent users were required to know the basic operations of PowerPoint. For each method, 25 frequent and 25 infrequent users were assigned the 50 figures at random. Each participant edited exactly one output, yielding one editing task for each figure–method combination. Equal representation of frequent and infrequent users balances experience across method groups. Reported times combine both groups. Participants were compensated per figure at prevailing market rates.

D.2 TASK MATERIALS AND EDITING PROTOCOL

Figure selection. We selected 50 figures with the highest information complexity from the 150-figure benchmark. For each sample, `gpt-5.6-sol` with `high` reasoning effort jointly assessed the input figure and its GPT Image 2.0 beautification in three independent blinded evaluations. Each evaluation assigned an information-complexity score in 0.5-point increments, with a maximum of 5. We ranked the samples by their mean scores and retained the top 50. All eight methods produced successfully rendered outputs for these samples, and all selected outputs could be imported and edited in PowerPoint.

Common reference and correction criteria. For each sample, outputs from all methods were edited to match the visual presentation of the same GPT Image 2.0 reference, including its layout and colors. The edited outputs were required to preserve the scientific content of the original input figure, including text, numbers, formulas, modules, connections, arrow directions, and containment relations. When the generated reference conflicted with the original input in content or logic, the original input determined the correction requirements. Icons could be replaced by visually similar or semantically equivalent alternatives. Participants standardized fonts within each figure, typically retaining the predominant font and correcting inconsistent instances. Exact reproduction of typography in the generated reference was not required.

Editing environment. Editing took place in Microsoft PowerPoint on Windows. SVG outputs were imported and converted to PowerPoint shapes before the timed task. Participants were not told which method produced their assigned output and received no preparatory demonstration or practice task. They could ungroup objects, modify existing elements, and delete and redraw content within PowerPoint. Participants could download icons from the web but could not use other editing software or AI assistance. Participants submitted both the edited PowerPoint file and its rendered image.

D.3 Acceptance and time measurement

Three reviewers with experience in scientific figure preparation or review independently inspected the PowerPoint file and rendered image against the correction criteria. Reviewers were not given method identities and made binary pass/fail judgments. A task was complete only when all three reviewers approved it. Otherwise, the participant continued editing and resubmitted the result.

Time was recorded automatically from the moment the experimenter opened the prepared PowerPoint file for the seated participant. The recorded duration included inspection, editing, ungrouping, saving, and all subsequent revision rounds. Timing paused during reviewer assessment and participant breaks and resumed when editing continued. Importing and converting the SVG before the task was excluded. There was no time limit, and all 400 tasks were completed with no participant withdrawals or record exclusions.

D.4 Results

Let t_{im} denote the cumulative active time until acceptance for figure i and method m. We report the arithmetic mean over the 50 editing tasks for each method:

$$\overline{T}_m = \frac{1}{50} \sum_{i=1}^{50} t_{im}. \tag{25}$$

FigStruct has the lowest mean editing time for each of the three controller configurations, ranging from 6 minutes 26 seconds to 9 minutes 15 seconds. These results characterize manual correction effort on the selected complex figures under a common visual target and acceptance protocol. The measured time reflects both the discrepancies in the initial output and the ease of editing it in PowerPoint.

Table 8: Mean active editing time until unanimous acceptance by three reviewers. Each method is evaluated on the same 50 figures, with one distinct participant per output and equal numbers of frequent and infrequent PowerPoint users. Times pool both groups and include all revision rounds, excluding reviewer waiting time and breaks. Lower is better.

Method	**Mean time (h:mm:ss)**
AutoFigure	0:45:19
Crafter	0:58:10
Direct (Gemma-4 31B)	2:26:34
Direct (GLM-5.3 Flash)	1:47:15
Direct (GPT-5.6 Terra)	1:55:03
FigStruct (Gemma-4 31B)	0:09:15
FigStruct (GLM-5.3 Flash)	0:07:48
FigStruct (GPT-5.6 Terra)	0:06:26

Post-editing efficiency. On the 50 selected complex figures, FigStruct reduces mean active editing time relative to the corresponding Direct configuration by 93.7%, 92.7%, and 94.4% for Gemma-4 31B, GLM-5.3 Flash, and GPT-5.6 Terra, respectively. This reduction is observed across all three controller configurations, supporting the practical value of the complete FigStruct workflow in reducing subsequent manual correction effort. All outputs must satisfy the same visual and content requirements and receive unanimous approval from three reviewers. The measured differences therefore reflect the active human effort required to bring generated outputs to a common acceptance standard. In scientific figure preparation, shorter post-editing times enable researchers to finalize figures with less manual correction and provide practical evidence that complements visual fidelity and structural editability metrics.

E Dedicated Case Studies

E.1 Case Study 1: Reconstructing an information-dense clinical workflow

Figure 6 compares the reference with FigStruct outputs using Gemma-4 31B, GLM-5.3 Flash, and GPT-5.6 Terra as the controller. The analysis focuses on local layout, connector clarity, and pictogram fidelity within the same dense three-stage workflow. Red boxes highlight selected local reconstruction discrepancies.

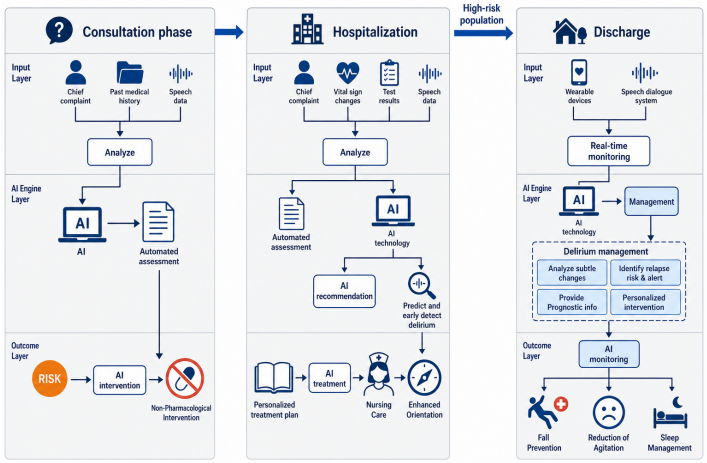
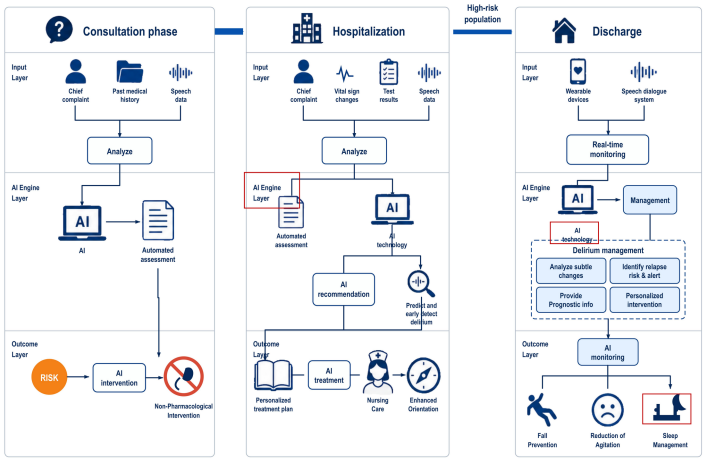
Case	Analysis
Original Figure 	**Dense reference.** Consultation, hospitalization, and discharge each contain input, AI-engine, and outcome layers. **Reconstruction challenge.** Short labels, branching connectors, a nested four-item management block, and small clinical pictograms must remain distinguishable. The main challenge is coordinating these local elements while retaining the three-column hierarchy and the flow between stages.
Gemma-4 31B 	**Local crowding.** The boxed hospitalization region places the assessment icon close to the layer heading. In the discharge panel, the “AI technology” label intersects the dashed management boundary. **Pictogram distortion.** The boxed sleep icon is compressed into a heavy, fused silhouette, reducing the separation between its components. The overall three-stage layout remains recognizable, but these local defects reduce readability and visual fidelity.

Figure 6: Case Study 1: FigStruct reconstructions of an information-dense clinical workflow with different controllers. The reference and Gemma-4 31B output are shown above; GLM-5.3 Flash and GPT-5.6 Terra continue on the next page. Red boxes highlight local discrepancies.

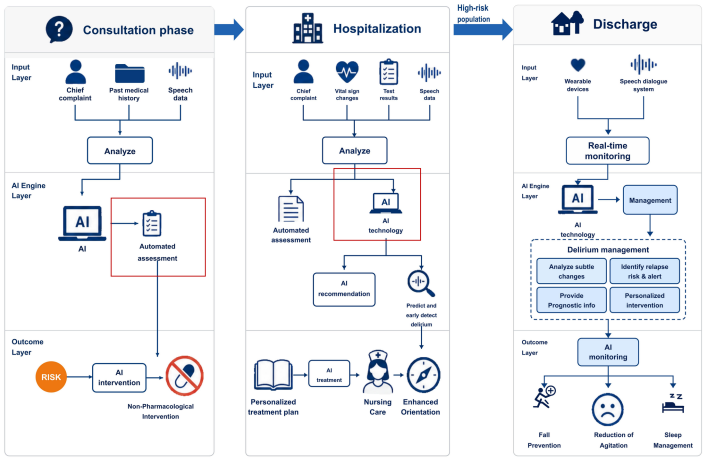
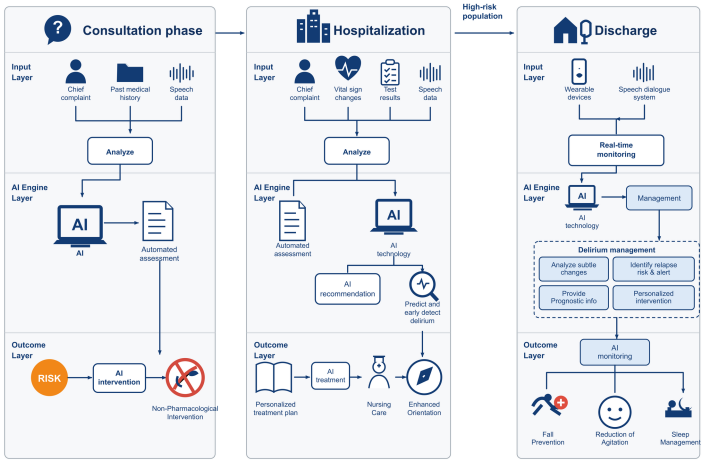
Case	Analysis
GLM-5.3 Flash 	**Content and logical structure.** The output largely retains graphical assets, text labels, and logical relationships. **Incomplete visual refinement.** In the highlighted regions, icons, labels, and connectors are not consistently aligned along a shared centerline. These local alignment inconsistencies weaken compositional balance and visual polish, despite otherwise adequate content preservation.
GPT-5.6 Terra 	**Asset and information preservation.** This output provides the most complete preservation of graphical assets and semantic information among the compared controllers. For icons not recovered by SAM 3.1, the controller attempts to redraw them. The overall visual presentation is also more coherent. **Minor residual artifacts.** Some pictograms are simplified in this dense diagram. Connections are retained, but unwanted arrowheads remain visible at a few junctions.

Figure 6 (continued). The GLM-5.3 Flash and GPT-5.6 Terra reconstructions illustrate differences in local layout and pictogram reconstruction.

Case-level observation. In this case, GPT-5.6 Terra retains the most complete set of graphical assets and semantic information among the three controllers. Its remaining pictogram simplifications and unwanted arrowheads distinguish asset recovery from accurate reconstruction of local details. These observations illustrate why information retention and connector quality are assessed separately.

E.2 Case Study 2: Spatial organization and composite graphical assets

Figure 7 examines a dense diagram of backbone architectures, self-supervised learning, and task adaptation using the same three FigStruct controllers. The comparison focuses on spatial clearance, the reconstruction of composite graphical assets, and the balance between connector regularity and legibility. Red boxes mark the regions discussed below.

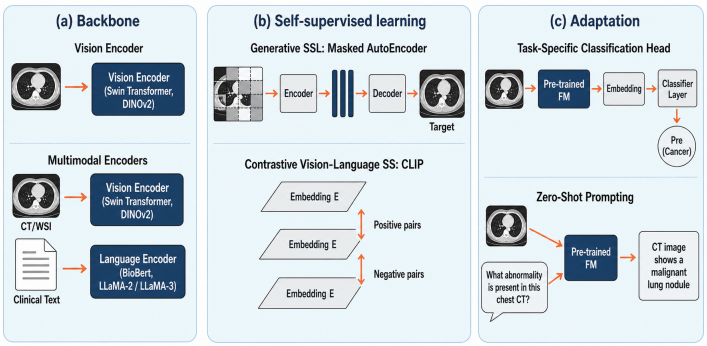
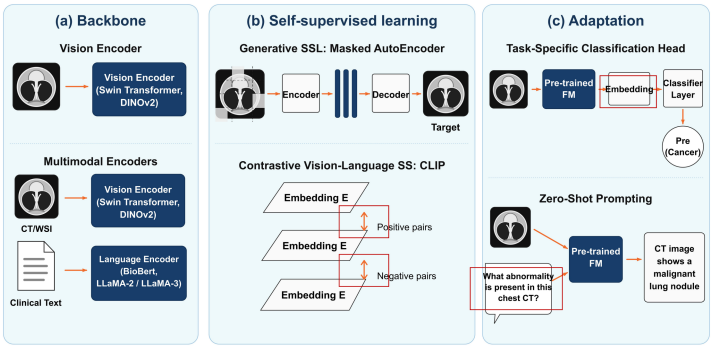
Case	Analysis
Original Figure 	**Dense reference.** Three panels combine encoder architectures, self-supervised learning, and task adaptation. **Reconstruction challenge.** Image assets, masking grids, multiline labels, and branching connectors require coordinated sizing and placement. Content and relationships must be preserved without crowding labels or connectors.
Gemma-4 31B 	**Limited spatial refinement.** Pair-relation arrows touch the embedding boundaries. Similar contact appears in the reference, but the reconstruction does not improve local clearance. **Text overflow.** The "Embedding" label and prompt-bubble text extend slightly beyond their containers, reflecting a mismatch between text size and available space.

Figure 7: Case Study 2: FigStruct reconstructions of a dense diagram combining learning workflows and composite graphical assets. The reference and Gemma-4 31B output are shown above; GLM-5.3 Flash and GPT-5.6 Terra continue on the next page. Red boxes mark the analyzed regions.

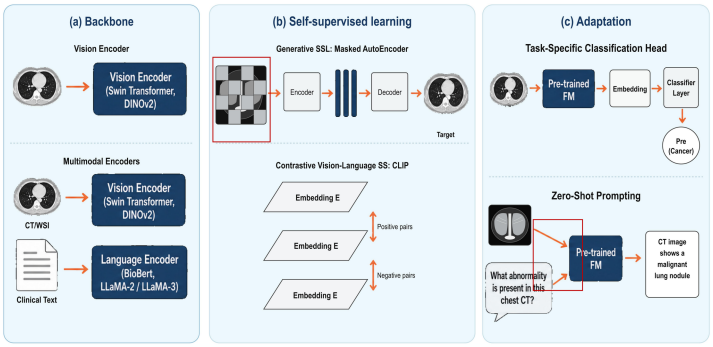
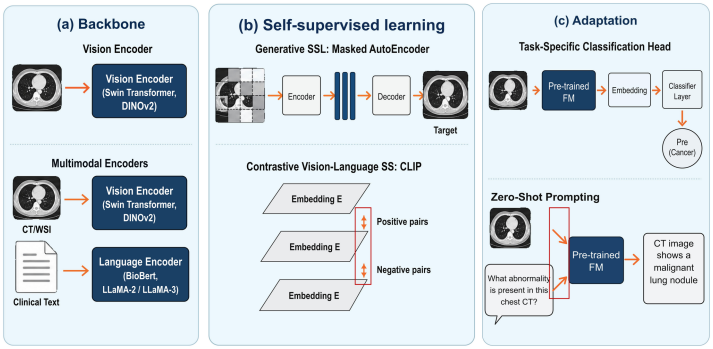
Case	Analysis
GLM-5.3 Flash 	**Semantic reconstruction.** A reused image asset is combined with a newly drawn masking grid to represent the masked input. This composition reflects the intended operation, but its rendering remains visually coarse. **Uneven connector geometry.** In the zero-shot branch, the two incoming arrows differ noticeably in length, weakening symmetry and visual balance.
GPT-5.6 Terra 	**Spatial consistency.** Arrow lengths and spacing are more regular, including the two incoming arrows in the zero-shot branch. Larger text improves readability relative to GLM. **Regularity and legibility.** The positive/negative-pair arrows have consistent lengths but are overly short, making their directionality harder to distinguish. This illustrates a trade-off between geometric regularity and local legibility.

Figure 7 (continued). The GLM-5.3 Flash and GPT-5.6 Terra outputs illustrate differences in composite-asset reconstruction, connector geometry, and local legibility.

Case-level observation. In this case, GPT-5.6 Terra produces more regular connector lengths and larger text than GLM-5.3 Flash. The short arrows in the positive/negative-pair branches also show that geometric regularity can reduce directional legibility. All three outputs retain local defects, making connector readability a useful complement to overall layout assessment.

F Comparison Case Analysis

The following four cases compare SVG reconstructions by AutoFigure, Crafter, and FigStruct from the same AI-generated image, shown as Non-Editable Input. All four FigStruct reconstructions use Gemma-4 31B as the controller and GPT-5.6 Luna for Global Refinement. Red boxes reproduce the author-provided annotations.

F.1 Comparison Case 1: Connector Geometry and Spatial Organization in a Dense Fusion Workflow

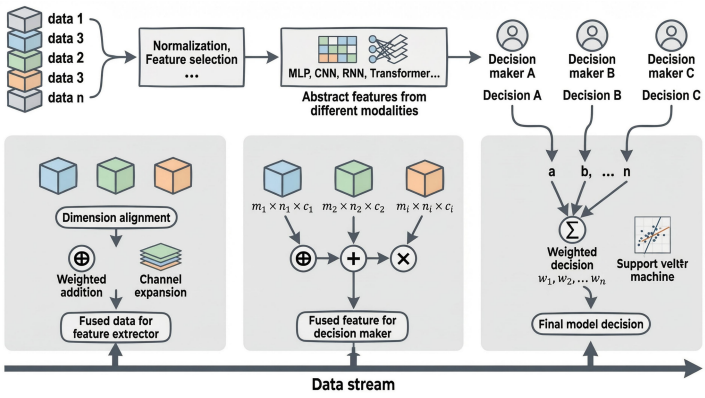
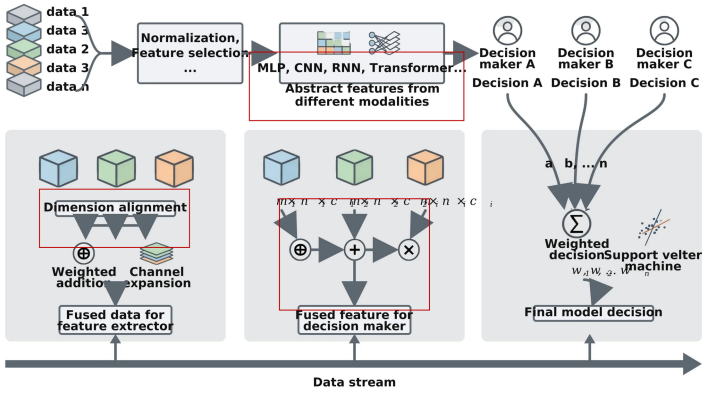
Case	Analysis
Non-Editable Input 	**Complex structure and reconstruction demands.** The input combines data processing, feature fusion, and decision fusion within one workflow, with dense labels, mathematical symbols, and branching and converging connections. Reconstructing these elements in SVG requires coordinated visual recognition, spatial reasoning, and code generation to preserve logical relationships while controlling element placement, arrow proportions, and local spacing.
AutoFigure 	**Disproportionate arrow geometry.** Across the three highlighted regions, thin arrow shafts are paired with excessively large arrowheads. The oversized heads intrude into neighboring text and symbols, causing local occlusion and disrupting visual balance. These defects indicate insufficient control over arrow proportions and their clearance from surrounding content in this reconstruction.

Figure 8: Comparison Case 1: Non-Editable Input and the AutoFigure SVG reconstruction. The Crafter and FigStruct reconstructions continue on the next page.

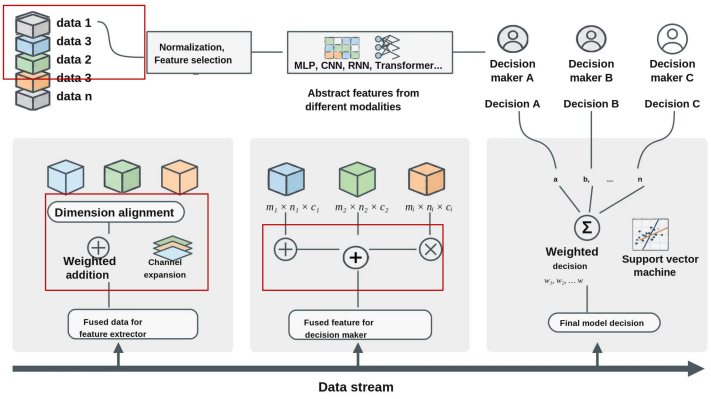
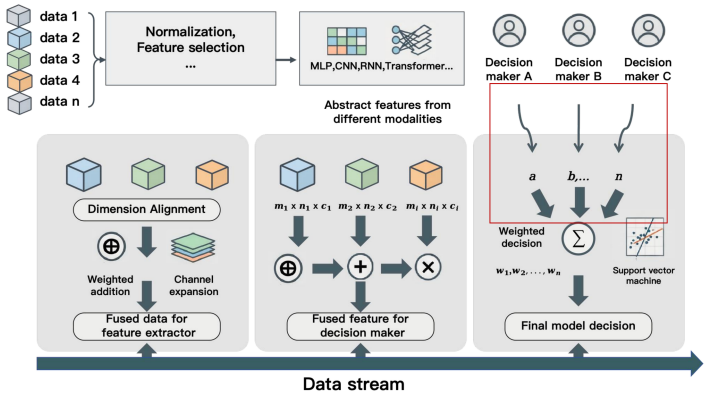
Case	Analysis
Crafter 	**Incomplete local structure and inconsistent alignment.** The grouping brace at the input is only partially reconstructed, some arrows are omitted, and local elements are not consistently aligned. Although the main modules remain recognizable, these defects weaken the clarity of grouping relationships and information flow, revealing limitations in fine-grained graphical completeness and spatial coordination in this case.
FigStruct 	**Coherent spatial organization.** The main modules are arranged with generally consistent spacing, keeping the workflow structure clear. The highlighted region nevertheless contains uneven arrow thickness and imbalanced visual weight, slightly reducing local aesthetic quality. This result shows effective spatial coordination while leaving room for improvement in connector styling consistency.

Figure 8 (continued). Crafter and FigStruct SVG reconstructions from the same Non-Editable Input.

Case-level observation. This case reveals three distinct patterns of local reconstruction error. AutoFigure struggles with arrow proportions and spatial clearance, affecting the legibility of neighboring text and symbols. Crafter exhibits incomplete preservation of graphical details and directional connections, together with local misalignment. FigStruct maintains more coherent spatial organization, with the highlighted issue primarily involving inconsistent arrow thickness. Together, these observations suggest that reconstructing dense diagrams in SVG requires coordination among visual recognition, relational understanding, spatial layout, and precise implementation in code. FigStruct demonstrates effective layout coordination in this case, while further refinement of connector geometry and styling remains necessary.

F.2 Comparison Case 2: Text Layout and Reference Fidelity in a Dense Taxonomy Diagram

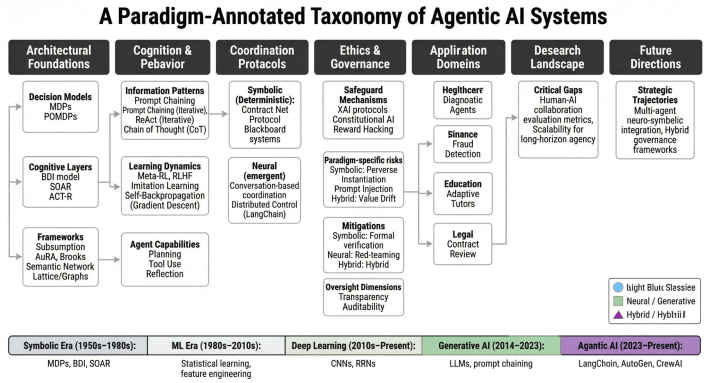
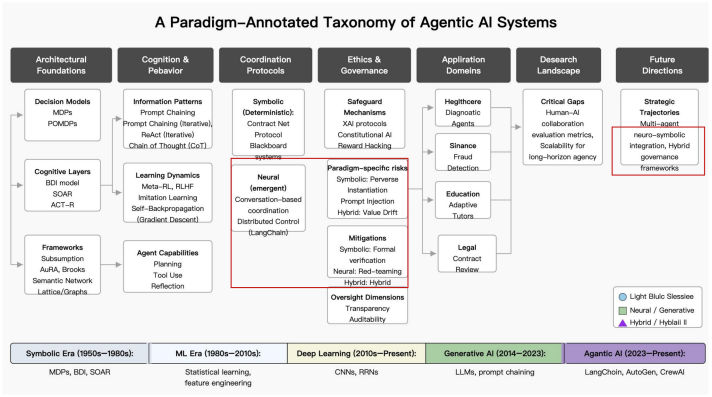
Case	Analysis
Non-Editable Input 	**Dense text and input ambiguity.** This taxonomy diagram contains multiple columns of text blocks and a compact legend, demanding careful control of text fitting, alignment, and spacing. Despite beautification with a capable GPT image-generation model, the lower-right legend still contains garbled text. This example highlights the difficulty of processing such text-dense diagrams. Visual beautification can introduce minor textual noise, creating ambiguity in content interpretation during subsequent SVG reconstruction.
AutoFigure 	**Text overflow.** In the highlighted regions, some text extends beyond its enclosing boxes, indicating insufficient coordination among text size, line wrapping, and container dimensions. These local overflows weaken the clarity of module boundaries, disrupt the orderliness of the dense, multi-column layout, and degrade the visual quality of the rendered output.

Figure 9: Comparison Case 2: Non-Editable Input and the AutoFigure SVG reconstruction. The Crafter and FigStruct reconstructions continue on the next page.

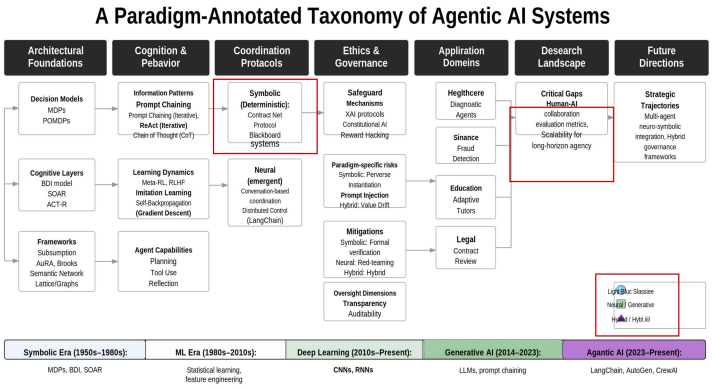
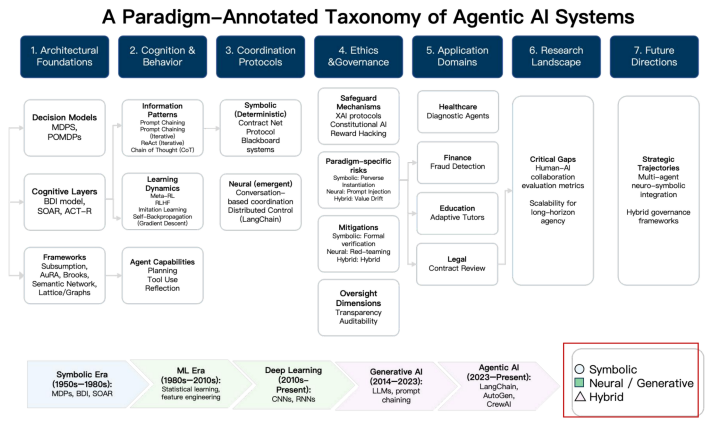
Case	Analysis
Crafter 	**Text overflow and legend misalignment.** Some text extends beyond its containers, while the lower-right legend places icons out of alignment with their corresponding labels. The former disrupts the organization of text blocks; the latter makes symbol–label associations less clear. These defects reveal limitations in text layout and the spatial coordination of associated elements in this case.
FigStruct 	**Orderly layout with interpretive revision.** The overall arrangement is regular, with well-coordinated placement of text within its containers. However, FigStruct appears to infer the intended categories of the garbled legend from context, producing semantically plausible labels. This improves legibility but introduces an interpretive modification of the reference; it does not constitute strict reproduction, and its semantic correctness cannot be established from the corrupted input alone.

Figure 9 (continued). Crafter and FigStruct SVG reconstructions from the same Non-Editable Input.

Case-level observation. This case highlights both the layout demands of text-dense diagrams and the fidelity challenges introduced by ambiguous inputs. AutoFigure exhibits insufficient coordination between text and container dimensions, while Crafter additionally misaligns legend symbols and labels. FigStruct provides more orderly overall formatting but introduces its own interpretation of ambiguous content. Reconstructing such diagrams therefore requires accurate text fitting and regular layout, together with a clear distinction between preserving visible content and completing meaning through contextual inference. A clearer presentation can coexist with departures from the reference, so improvements in readability and fidelity should be assessed separately.

F.3 Comparison Case 3: Icon Reconstruction and Spatial Coordination in a Complex Layout

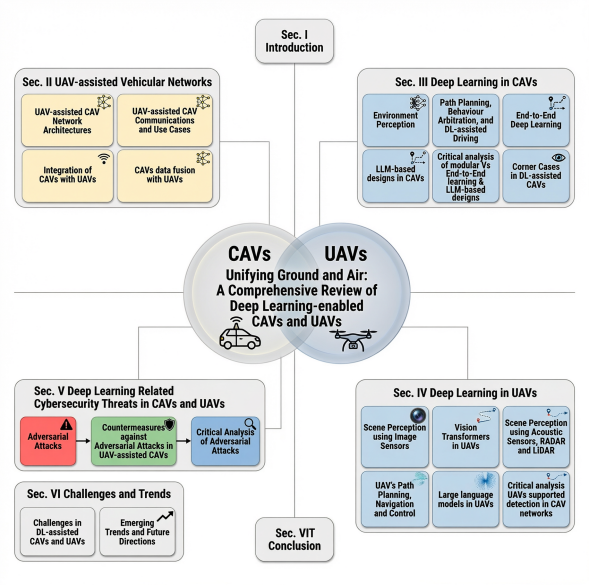
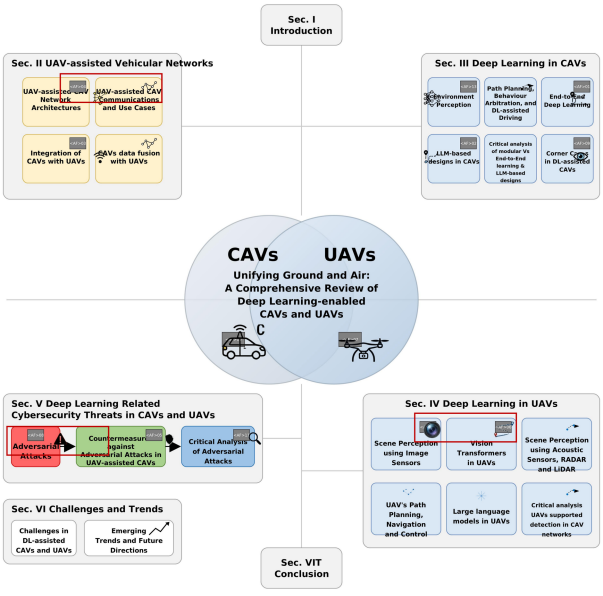
Case	Analysis
Non-Editable Input 	**Complex layout with distributed icons.** The input combines nested modules, icons distributed across multiple regions, and connections between these regions. Reconstruction must preserve the module hierarchy, icon–content associations, and connection structure while coordinating spacing among icons, text, and boundaries. This places substantial demands on visual recognition, spatial reasoning, and implementation in code.
AutoFigure 	**Icon placeholders and inappropriate placement.** Multiple icon locations retain placeholders, while some correctly reconstructed icons are positioned inappropriately within their panels. These defects reduce the completeness and spatial consistency of icon presentation, introduce local visual clutter, and weaken the coherence and overall visual quality of the rendered diagram.

Figure 10: Comparison Case 3: Non-Editable Input and the AutoFigure SVG reconstruction. The Crafter and FigStruct reconstructions continue on the next page.

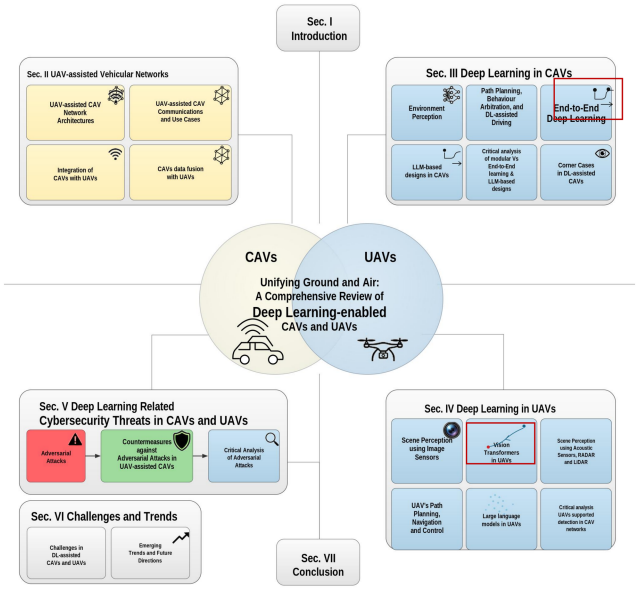
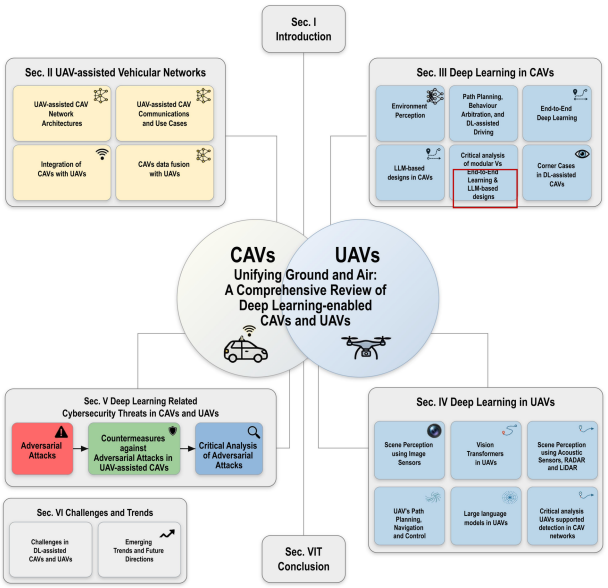
Case	Analysis
Crafter 	**Low-quality substitute icons and icon–text overlap.** When original icons are not successfully obtained, Crafter draws substitutes, some of which are coarse and simplified. Certain redrawn icons also intrude into adjacent text regions, causing icon–text overlap. These artifacts reduce both icon fidelity and text legibility, revealing limitations in substitute drawing and local spatial coordination.
FigStruct 	**Accurate placement with limited local spacing refinement.** Icons are appropriately positioned, and the modules form a clear, orderly layout. However, the highlighted text still touches the lower boundary of its container, retaining a local spacing defect already present in the Non-Editable Input. This result demonstrates effective icon placement and overall organization, while leaving an inherited layout defect insufficiently refined.

Figure 10 (continued). Crafter and FigStruct SVG reconstructions from the same Non-Editable Input.

Case-level observation. This case highlights the need to coordinate icon completeness, drawing quality, and spatial relationships in complex layouts. AutoFigure primarily exhibits residual placeholders and inappropriate icon placement. Crafter struggles with substitute drawing and icon–text separation when original icons are unavailable. FigStruct maintains appropriate icon placement and clear organization but retains an inherited spacing defect. Together, these observations suggest that reliable icon recovery or redrawing must be coupled with precise local layout control, and that preserving the reference structure should also accommodate refinement of existing crowding and boundary-contact artifacts.

F.4 Comparison Case 4: Panel Boundaries and Structural Fidelity in a Composite Figure

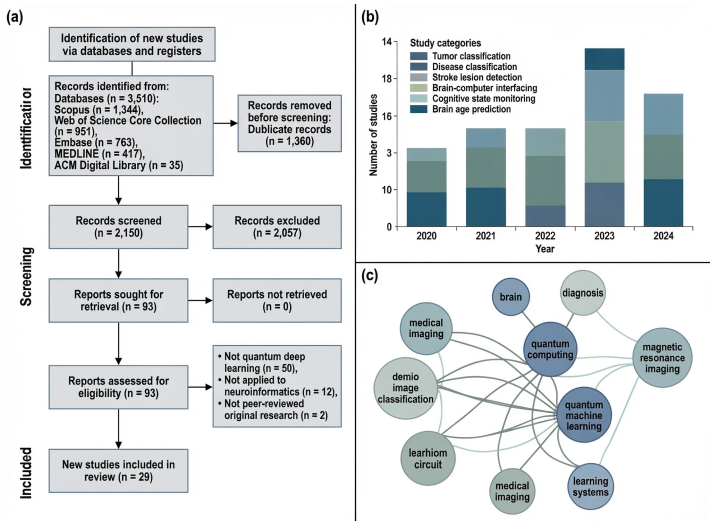
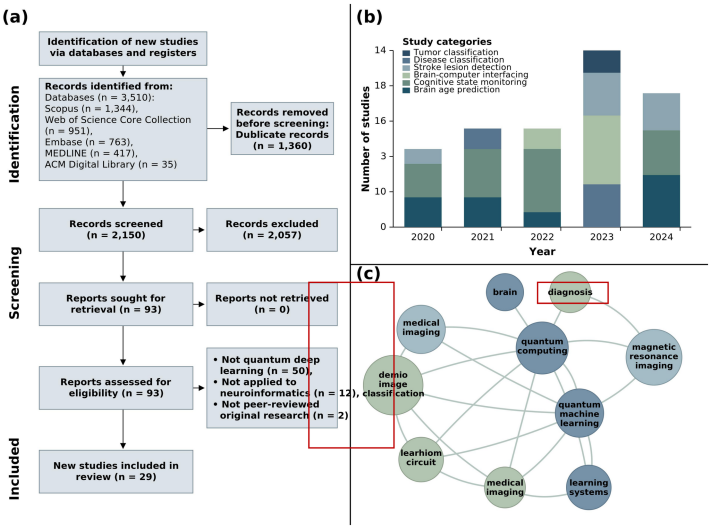
Case	Analysis
Non-Editable Input 	**Composition of multiple visual structures.** The input combines a flowchart, a stacked bar chart, and a node-link diagram on a shared canvas, with different layout conventions and connection patterns across panels. Reconstruction requires coordinating the overall panel arrangement while accurately handling local text, arrows, axes, and node connections, placing substantial demands on spatial planning and SVG implementation.
AutoFigure 	**Text overflow and cross-panel interference.** Some text extends beyond its enclosing containers and even crosses the divider into a neighboring panel. These defects disrupt local text layout and weaken the visual separation between subfigures, indicating insufficient coordination between text fitting and panel boundaries and reducing the overall quality of the rendered output.

Figure 11: Comparison Case 4: Non-Editable Input and the AutoFigure SVG reconstruction. The Crafter and FigStruct reconstructions continue on the next page.

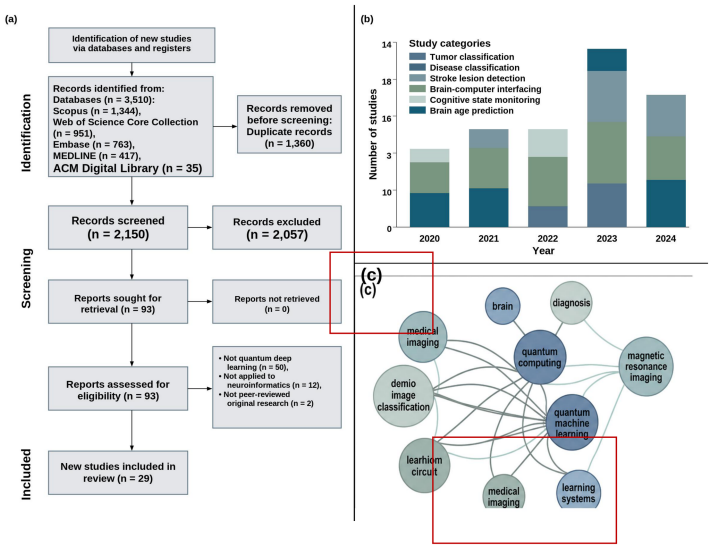
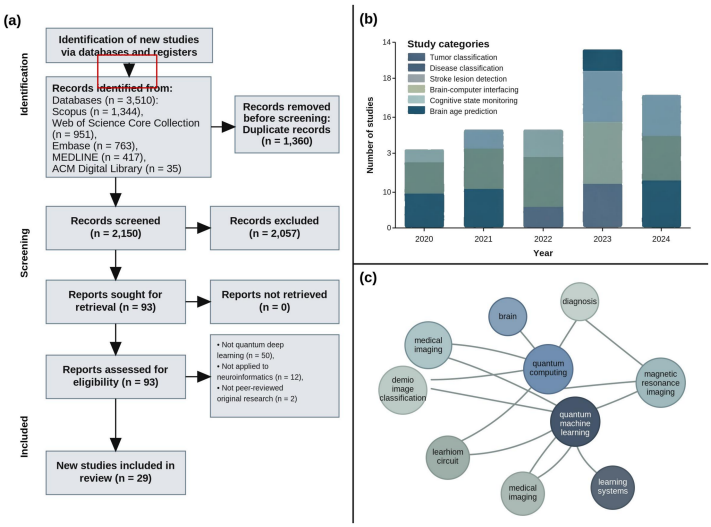
Case	Analysis
Crafter 	**Duplicated panel label and bottom clipping.** The lower-right panel contains a duplicated "(c)" label. In addition, a mismatch between the SVG rendering dimensions and the content extent clips portions of the bottom nodes in the network diagram. These defects reveal limitations in panel annotation management and rendering-boundary control, compromising presentation consistency and completeness.
FigStruct 	**Orderly composition with local geometric and relational deviations.** The panels are arranged regularly with clear visual separation. However, one arrowhead in the flowchart is oversized, creating a local imbalance in visual weight. Connections in the lower-right panel "(c)" also show a tendency toward simplification relative to the reference. The overall layout is therefore well organized, while arrow proportions and preservation of fine-grained connectivity remain areas for improvement.

Figure 11 (continued). Crafter and FigStruct SVG reconstructions from the same Non-Editable Input.

Case-level observation. This case highlights the need to jointly control panel boundaries, rendering extent, and the internal structures of different diagram types. AutoFigure primarily exhibits text overflow and cross-panel interference. Crafter introduces duplicated panel labeling and bottom clipping, compromising presentation completeness. FigStruct achieves an orderly overall layout but retains an arrow-scale imbalance and simplifies some local connections. Together, these observations indicate that effective visual organization must be accompanied by complete content rendering and relational fidelity.

G METHOD DETAILS AND EXECUTION TRACES

G.1 OBJECT RECORDS AND STRUCTURED CORRECTION PLANS

Object records and bindings to SVG elements. The object hierarchy described in the main text organizes the current SVG into modules, nested groups, and individual objects. Each node records its identifier, type, spatial region, and content, with bindings to the SVG elements or raster assets that implement it. Node status records its lifecycle state and, where applicable, asset protection status. Parent–child relationships provide structural context, spatial regions help locate the relevant visual content, and bindings connect diagnoses to specific editable elements. A *target* is a module or subregion selected for joint revision, providing context for the objects being repaired.

From visual diagnoses to structured correction plans. Error Review compares the current SVG render with the design image I_{AI} to identify content and presentation errors. Region Localization then uses the object hierarchy to associate each actionable diagnosis with the affected targets and objects. The resulting correction plan records the error description, expected correction, target and object identifiers, relevant hierarchy context, and recommended repair scope. Repeated diagnoses of the same target are consolidated before repair. The design image provides the content reference for these correction plans.

This information distinguishes the context needed for repair from the selected revision scope. For a truncated label, the worker inspects its enclosing group and neighboring elements to assess available space and layout. Object identifiers in the plan specify the text object to correct. When a diagnosis concerns the layout or internal connections of an entire module, the plan can recommend coordinated revision of that target. The following subsection describes fragment replacement, spatial boundaries, and preservation constraints.

G.2 LOCAL SVG REPAIR PROTOCOL

Repair scope and fragment replacement. Object-focused correction and target-level revision specify different edit scopes. Object-focused correction addresses selected text, shapes, or connectors, whereas target-level revision permits coordinated changes to the internal layout, object organization, and connections of the selected region. In both cases, the worker receives the complete SVG fragment rooted at the selected target. It returns a complete native SVG replacement retaining the same target identifier. The complete fragment supplies layout context and a consistent replacement unit, while the correction plan determines which objects can change.

Fixed rendered boundary. At worker initialization, the system renders the current target, measures its actual outer bounding box, and fixes this region as B_T from the main text. The boundary records the left, top, right, and bottom edges in rendered coordinates and remains unchanged throughout subsequent review and repair by that worker. A new candidate cannot resolve overflow by enlarging the boundary or replacing B_T with its own bounding box. Both object-focused correction and target-level revision must remain within this original boundary, including when the entire internal layout is reorganized.

Identifier and asset preservation. Repair preserves the current target identifier, unselected target subtrees, and the identities and content of protected raster assets. The system retains asset identifiers, bindings, and bitmap content according to the asset registry. Asset positions and sizes may change within the current edit scope and fixed boundary. Internal module connections are handled with their corresponding targets, while cross-module connectors are revised separately after the endpoint modules are established.

Candidate review and execution records. Each candidate replacement fragment is rendered to inspect the correction and the actual boundaries of its target and objects. Overflow checks use the rendered result and therefore account for visible extents caused by features such as strokes. A candidate that passes visual review must also pass the boundary check. Detected overflow is recorded to guide subsequent repair. For each attempt, the system retains the complete replacement fragment, rendered target and object boundaries, overflow report, and review decision, allowing the local repair process to be traced and audited.

G.3 Merge, Hierarchy Reconciliation, and Validation

Fragment merging and conflict handling. After the parallel workers complete local repair, Merge inserts accepted replacement fragments using their unchanged target identifiers, while retaining unselected target subtrees and protected asset bindings. When selected targets are nested, child replacements are incorporated into the corresponding parent fragment before insertion. Identifier, ancestry, or SVG-reference conflicts abort the merge if unresolved, preserving accepted changes.

Updating the hierarchy from the current SVG. Reconcile reparses the updated SVG and updates the existing object hierarchy from the object identifiers, parent–child relationships, and element bindings actually present in it. Surviving objects retain their identifiers, new objects are registered, and changes in group membership are reflected in the hierarchy records. Objects present in the previous records but absent from the updated SVG are flagged for content review.

Content review of removed objects. The disappearance of an object identifier alone does not establish whether its content has been handled correctly. Using the design image and current render, review classifies each flagged object as *legitimately removed*, *replaced*, *still required*, or *uncertain*. For example, an object is classified as *replaced* when its original identifier disappears but its content remains represented by other objects or structures. If information required by the design image is missing from the current render, the object is classified as *still required*, and the necessary follow-up correction is recorded. Uncertain cases remain recorded without independently blocking correction.

Validation and content review. Programmatic validation checks XML parseability, target-identifier retention, and identifier uniqueness. It also checks protected asset bindings and bitmap content, and verifies that SVG references resolve to existing elements. Together with the fixed-boundary checks described in the preceding subsection, these checks constrain candidate updates. Content review checks information against the design image and judges whether structural changes satisfy the correction plan. Unresolved parsing, identifier, ancestry, or reference errors reject the affected update. After merging and hierarchy reconciliation, the new SVG render and object tree return together to Agent Review for the next inspection round.

Output checks after Global Refinement. The pipeline ends after checking that the final SVG from Global Refinement parses as XML and renders successfully as a PNG image.

G.4 Optional Region Regeneration and Stage-Specific Constraints

Optional region regeneration. The implementation retains a region-regeneration fallback for severe logical errors or garbled content that cannot be adequately corrected through native SVG edits. The controller writes a revised generation prompt for the affected region to address the diagnosed content errors, requesting consistency with the colors, layout, and visual style of the complete figure. The generated region is pasted into the corresponding reconstruction context and subsequently converted into replacement SVG content. A local composite reference may guide this reconstruction, while the style reference supplied to Global Refinement remains the complete design image I_{AI} produced in Step 1.

Paste regions and node preservation. The diagnosed errors determine whether the model regenerates only the module interior or also its title and frame. For interior-only regeneration, the generated content is pasted back into the module interior, retaining the original native SVG frame and title. When the regeneration scope includes the title and frame, the generated region containing these elements is pasted back into the corresponding area and converted into replacement SVG content. The paste region is determined by the regeneration scope selected by the model. Old nodes fully contained in the paste region are superseded by the reconstruction, while neighboring nodes that merely intersect the region are retained. Cross-module connector paths are revised after the endpoint modules are established, using their updated positions and boundaries.

Generation attempts and evaluation use. Each affected region permits at most two regeneration attempts. In the main SVG reconstruction evaluation using 150 GPT Image 2.0 designs, repairs were

performed through SVG edits, and the region-regeneration fallback was not triggered. The SVG-stage costs reported for this configuration therefore contain no region-regeneration charges. We additionally analyzed the error types and corresponding repair requirements of 150 design images generated by Qwen Image 2.0 Pro. All 150 inputs ultimately yielded successfully rendered final SVGs. The mean number of region-regeneration calls was approximately 1.3 per output, calculated across all 150 inputs, including those that required no region regeneration. At an API cost of approximately USD 0.074 per call, the estimated additional cost of region regeneration was USD 0.096 per output for this Qwen-based reconstruction configuration.

Stage-specific edit scopes. The stages impose different spatial and preservation constraints. Structure conversion records existing SVG elements and their relationships. Local native SVG repair retains the initial boundary B_T, while region regeneration uses a separately determined paste region. Global Refinement coordinates layout across local boundaries. Table 9 summarizes these distinctions.

Table 9: Edit scopes and preservation constraints at each stage. The model selects either interior-only or whole-module regeneration when invoking the optional fallback. This fallback was not triggered in the reported 150-figure evaluation using GPT Image 2.0 designs.

Stage or operation	**Edit scope and spatial constraint**	**Preservation requirements**
SVG Structure Conversion	Build module, group, and object records for existing elements, registering their regions and bindings.	Retain existing SVG content. Construct the hierarchy once and update it subsequently.
Object-focused SVG correction	Modify selected objects, keeping the repaired result within the fixed B_T.	Retain target identifiers, unselected target subtrees, and protected assets.
Target-level SVG revision	Revise object organization, internal layout, and connections within the same B_T.	Retain target identifiers, unselected target subtrees, and protected assets; update object records accordingly.
Whole-module regeneration	The model may choose to regenerate the module content together with its title and frame.	Supersede old nodes fully contained in the paste region; retain neighboring nodes that merely intersect it.
Interior regeneration	The model may choose to regenerate only interior content, keeping the paste region inside the retained frame and title area.	Retain the native SVG frame and title, as well as neighboring nodes that intersect the paste region.
Cross-module connector revision	Revise paths after the endpoint modules are established, using their current identifiers and boundaries.	Retain endpoint module content. Internal connectors remain with their corresponding modules.
Global Refinement	Adjust layout, typography, paths, grouping, asset placement, and the canvas across local target boundaries.	Preserve scientific content, assets, connector validity, and the design aspect ratio; use the fixed I_{AI} as the style reference.

G.5 Recorded Execution Traces

The following three cases illustrate object-focused repair, adjustment of the repair scope, and hierarchy updates with content review after structural edits.

Object-focused label repair. In sample `406_Fig_01_Thi`, the label "Dynamical Systems Neural Networks" was truncated to "Dynamical Systems Neural N". The diagnosis localized the error to text object `classical_sub2_text` within target `branch_classical_mechanics` and recommended an object-focused SVG correction. The enclosing group, background, and list elements supplied layout context, and the asset registry protected icon `icon_001` from modification.

The worker fixed B_T at the start of repair and returned a complete replacement target fragment, reflowing the full label into two SVG `tspan` elements within the selected text object. Re-rendering

detected no boundary overflow, reconciliation retained all existing object identifiers, and the candidate passed its first review without an additional repair round.

Layout revision and boundary repair. In sample `200_Fig_04_Tec`, the real-time flight-control module contained a spurious "4." in its title and incorrectly arranged three control functions in a linear sequence. The diagnosis selected the title, node group, and connector group of the module, requiring "Attitude control" and "Emergency control" to appear above "Flight management" with triangular directed connections.

Within the fixed boundary $B_T = [153, 673, 1383, 830]$, the worker revised the module by removing the spurious numbering, reorganizing the layout, and restoring the three directed connections. The first visual review accepted the corrected title, layout, and connections, but the boundary check found that the stroke of background object `mod_ctrl_bg` extended beyond the nominal rectangle, causing one-pixel overflow on each side.

The subsequent diagnosis localized the remaining issue to this background object and recommended an object-focused repair. The worker inset the background rectangle while retaining the corrected title, nodes, internal connections, and protected icon `icon_006`, with the original B_T unchanged. The second render contained no overflow and passed review. This case shows how successive reviews change the selected objects while preserving the target identifier and spatial boundary.

Reconciliation after module removal. In another target, `branch_quantum_ml`, from sample `406_Fig_01_Thi`, the diagnosis identified a spurious intermediate module labeled "Quantum Learning Algorithms". Before revision, the algorithm list was nested within the intermediate module. The target-level revision removed the redundant group and its background and text objects, then reparented the algorithm list directly under the target.

The updated local hierarchy contained nine objects instead of twelve, with no new object identifiers introduced. The retained header group, icon `icon_004`, algorithm-list members, and internal connector kept their existing identifiers and bindings. The hierarchy update flagged the three removed objects for review to determine whether their removal was justified.

Review confirmed that the required algorithm-list content remained represented and that the removed elements formed a redundant intermediate structure. The three removed objects were therefore classified as *legitimately removed* rather than *replaced*. The internal connector had also been revised to connect the retained header directly to the algorithm list, and the boundary check detected no overflow. The initial replacement for this target passed its first review without additional repair. Its nine objects formed the updated registry, while the removed objects and corresponding review decisions remained in the execution records. The two accepted fragments were merged after the independent classical-mechanics branch repair also completed.

G.6 Experimental and Reproducibility Settings

Crop and retrieval settings. The controller localizes regions on a 6×6 grid and may inspect up to four native-resolution crops. For each identified visual issue, one retrieval round considers at most two papers and four figures per paper, retaining up to six references for subsequent design planning.

Models and generation settings. The default controller is Gemma 4 31B IT, with temperature 0.3 and a maximum output length of 40,000 tokens. Local inference uses two NVIDIA H100 GPUs, each with 80 GB of memory. Image-generation configurations include GPT Image 2.0 and Qwen Image 2.0 Pro. The main SVG reconstruction comparisons use GPT Image 2.0 design images as inputs. The seed is set to 42 where supported, and the closest available output aspect ratio to the source is selected. Global Refinement uses GPT-5.6 Luna.

Concurrency, correction rounds, and retries. Step 2 performs at most two outer correction rounds. The repair-worker concurrency parameter defaults to three and can be configured up to five. Higher concurrency can overload the local small-model inference service when its throughput limit is reached, motivating the default setting of three workers. The concurrency limit does not cap the number of issues identified and recorded in each review round. Global Refinement performs one

pass per input figure. Failed API requests are retried until an output is obtained, independently of the two-round correction limit.

Runtime measurement. SVG reconstruction runtime measures the elapsed wall-clock time for Step 2 and Step 3 through final SVG rendering, averaged over all 150 inputs. Under the default configuration, these stages use Gemma 4 31B IT and GPT-5.6 Luna, respectively. For every compared method, elapsed time includes failed API requests, retries, and reconnection waits until the final output is obtained. Step 2 timing reflects the actual parallel execution of its repair workers.

Run records and cost accounting. Each run records the model configuration, input and output identifiers, random seed where supported, selected resolution, and stage status. Retrieval records also retain queries, candidate paper and figure identifiers, screening decisions, selected references, and timestamps. Costs are reported in USD and include charges for all billable calls, including failed calls and retries, within the measured stage until the final output is obtained for each input. The mean API cost is the total of these charges across all 150 benchmark inputs divided by 150. The fraction of inputs affected by generation errors is reported separately over all 150 benchmark inputs, counting each affected input once. Evaluation expenses and local hardware and electricity costs are excluded. Detailed accounting follows the appendix on generation cost and error analysis.