

A Benchmark for the Generation and Evaluation of Scientific Architecture Diagrams in HTML/SVG

Kyeongman Park¹ Kang-il Lee¹ Janghoon Han² Yerin Hwang¹
Taegwan Kang² Minwoo Lee² Changho Lee² Kyomin Jung¹

¹Seoul National University ²LG AI Research

{zzangmane, 4bkang, dpfls589, kjung}@snu.ac.kr

{janghoon.han, taegwan93.kang, minwoo.lee, changho.lee}@lgresearch.ai

Abstract

While Large Language Models (LLMs) can now automate many parts of scientific writing, architectural figure generation faces a critical modality bottleneck because image outputs are inherently difficult to edit, verify, and integrate into text-native LLM workflows. To enable fully text-driven synthesis, we propose a new task: generating architectural figures as pure HTML/SVG source code. Furthermore, we introduce FigureCode, a novel benchmark that evaluates structural fidelity and visual properties through code-based evaluation. Our benchmark improves agreement with human judgments by 0.35 Kendall's τ over prior VLM-based methods. Moreover, the figure-as-code paradigm improves edit-instruction success rate by 30% and retrieval F1 score by 25.3% over baselines, while achieving 66.1% WCAG/ARIA accessibility coverage. Project link is [here](#).

1 Introduction

Recently, large language models (LLMs) have begun to automate scientific paper writing, including ideation, drafting, and experiments (Lu et al., 2024; Schmidgall et al., 2025; Starace et al., 2025). However, even though architectural figures are often the most important components in a paper as they visually summarize the paper's main contribution (Tufte, 2001; Munzner, 2014; Pang et al., 2025; Huang et al., 2026), the *automatic generation of architectural figures* still remains a challenge.

Current approaches generate architectural figures using text-to-image (T2I) models (Ramesh et al., 2022; Saharia et al., 2022; Betker et al., 2023; Esser et al., 2024; Zhu et al., 2026), whose outputs are pixel grids. However, these pixel-based outputs often lack *editability* (Meng et al., 2022; Hertz et al., 2023; Brooks et al., 2023) and *verifiability* (Yang et al., 2025; Wu et al., 2025; Zhao et al., 2025; Wang et al., 2024; Hwang et al., 2025).

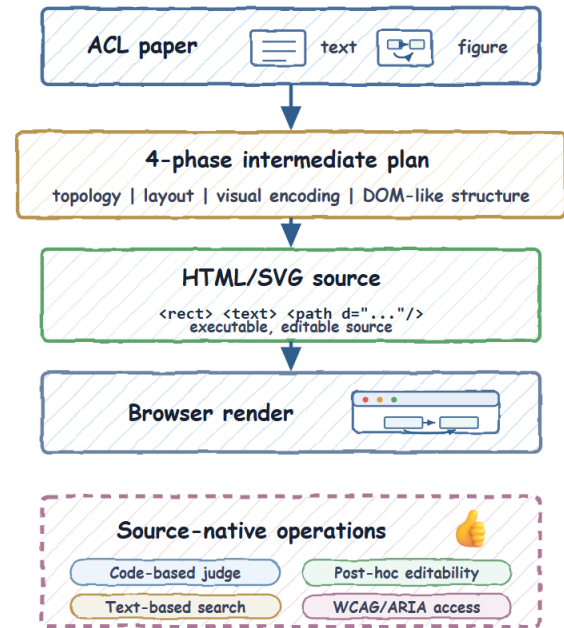


Figure 1: Overview of our benchmark, FIGURECODE. We start from an ACL paper and its figure, reverse-engineer a four-phase intermediate plan, reconstruct HTML/SVG source, and use the source for evaluation and generation. This figure is generated by our proposed method and is written entirely in HTML/SVG.

Furthermore, image generation hinders fully end-to-end paper automation by LLM agents, because images are not directly optimizable as text.

To address these limitations, we propose a novel task: architectural figure generation based on pure text. Specifically, the model generates HTML/SVG source code that can be rendered into a figure (Beltramelli, 2017; Laurençon et al., 2024; Si et al., 2025; Wu et al., 2025; Zhao et al., 2025; Yang et al., 2025). In this task, every component of a figure becomes an explicit text token that can be verified against the paper's claims. Moreover, the code is directly editable through textual revisions. Finally, the output is in the same modality as LLM agents, which enables fully end-to-end paper automation.

However, existing benchmarks are ill-suited for

evaluating this code-centric paradigm. Current formats either restrict outputs to rigid, domain-specific languages (Huang et al., 2026) or rely strictly on image-based evaluation via vision-language models (VLMs) (Zhu et al., 2026), which is severely decorrelated from human judgment (Zheng et al., 2023; Hwang et al., 2025).

To establish a reliable evaluation for figure generation, we introduce **FigureCode**, a novel benchmark for *figure-as-code* generation and evaluation. We collect 1,369 architectural figures from ACL-venue papers (2024–2025) and transform them into HTML/SVG source code using LLMs and human-in-the-loop processes. We then reverse-engineer intermediate plans for transforming the paper’s text into source code (Wei et al., 2022; Zhou et al., 2023; Chen et al., 2023; Si et al., 2025), constructing a triplet dataset (paper, intermediate plans, code) which can be used for training and evaluation. We also introduce a holistic evaluation framework covering paper-grounded fidelity and visual properties.

Surprisingly, FigureCode improves agreement with human judgments by 0.35, edit success by 30%, and retrieval F1 by 25.3%, while achieving 66.1% WCAG/ARIA coverage and competitive generation performance, suggesting that architectural figure generation can be reframed as a reliable, editable, and LLM-native code-generation task.

2 FigureCode Benchmark

2.1 Task Definition

We introduce a novel task, *figure-as-code*, which aims to generate an architectural figure as pure source code derived from a paper’s content. Given the paper text T , the system outputs an HTML/SVG document H that can be rendered into a figure via $I = \text{render}(H)$ using a rendering engine in web browsers. Unlike T2I generation, the target object is not merely the rendered bitmap but the source code itself. This ensures that all components remain inspectable and verifiable as text. A detailed comparison with related tasks is given in Appendix D. We do not use constrained Domain-Specific Language (DSL) such as draw.io (Huang et al., 2026) because architectural figures in real papers often contain graphically rich elements, nested groups, and free-form annotations. In contrast, HTML/SVG offers much more powerful expression (World Wide Web Consortium, 2018; Chen et al., 2025; Si et al., 2025).

2.2 Dataset Construction

FIGURECODE contains 1,369 architectural figures collected from ACL-venue papers published in 2024–2025, to maintain reasonable visual complexity. For each paper, we extract the target figure image and its corresponding text. To curate the dataset efficiently, we employ an LLM-as-a-judge framework (Zheng et al., 2023) to filter out statistical charts, graphs, or tables focused primarily on numerical data, as well as trivial samples such as simple text boxes. We additionally preserve the rendered images and metadata necessary for downstream evaluation.

2.2.1 HTML/SVG Reconstruction

Next, we reconstruct each selected figure into browser-renderable HTML/SVG source code. We utilize GEMINI-3.5-Flash (Gemini Team, 2023) to generate an initial HTML/SVG draft based on the paper text and the original figure. Crucially, we convert absolute pixel coordinates from the initial drafts into structural and relative responsive units (e.g., Flexbox, CSS grid) to emphasize layout hierarchy and semantic grouping rather than exact coordinates.

2.2.2 Intermediate Plan Annotation

For every curated figure and source code, we also construct a four-phase intermediate plan that explains how the paper should be converted into source code. The phases are: (1) extracting the topology of components and relations from the paper, (2) choosing the layout and grouping structure, (3) assigning visual encodings such as size, color, typography, and connector style, and (4) writing a DOM-like structural representation. This plan explains which parts of the figure are paper-grounded and which parts are design choices. To ensure the reliability of this intermediate representation, we verify that GEMINI-3.5-Flash can faithfully reconstruct the original HTML/SVG code when conditioned on this plan. Consequently, each data point is formalized as a triplet (T, P, H) , where T denotes the paper text, P represents the intermediate plan, and H signifies the final HTML/SVG source code.

2.2.3 Human Verification

Finally, we conduct a meticulous manual quality check on the ground-truth source code H . Ph.D. students rigorously inspect whether the generated code fully preserves the major components, inter-

element relations, textual labels, structural hierarchy, and visual grouping of the original human-authored figure. This verification process yields the final FIGURECODE dataset, consisting of 1,369 high-quality, fully-reconstructable figures and their (T, P, H) triplets.

2.3 Evaluation Framework

FIGURECODE evaluates both the generated source and the rendered figure. Our main evaluator is a code-grounded source judge that receives the paper text and raw HTML/SVG source, subsequently assessing paper-grounded fidelity and rendering quality. Every score must be justified with HTML- and paper-evidence snippets (Appendix C). For the main result table, we report six representative sub-metrics grounded in HCI and information-visualization principles (Tufte, 2001; Munzner, 2014). 1. *hallucination* : no components or formulas appear that are not in the paper; 2. *relation correctness* : inter-component relations match those described in the paper; 3. *label accuracy* : component and method names are labeled as in the paper. 4. *edge crossing* : connection lines avoid crossing each other; 5. *visual hierarchy* : main components stand out in size, weight, or position; 6. *proper margin* : sufficient spacing between elements. To make the evaluation comparable across modalities, T2I outputs are first converted into HTML/SVG through the same process as dataset generation, and then the same judge is applied.

3 Experiments

3.1 Human Correlation Experiments

We first test whether evaluating a generated figure through its source code matches human judgment better than evaluating it via an LVLM. We use the 60 human-scored figures collected for the correlation analysis, with each figure rated by 10 annotators on a 5 Likert scale. Further details on the human evaluation are provided in Section B. Each output is scored by two automatic judges: an LVLM judge that receives the rendered figure and paper text, and a code-grounded source judge that receives the raw HTML/SVG source and paper text. The correlation of each judge with human scores is computed using Kendall’s τ for the six evaluation criteria used in the main result table.

Tab. 1 shows that the source-based judge improves agreement with human ratings on every reported criterion. The largest gains appear on

Metric	LVLM	Code-judge
Halluc.	+0.132	+0.766
Rel. corr.	+0.083	+0.458
Label acc.	+0.250	+0.715
Edge cross.	-0.121	+0.308
Hierarchy	+0.211	+0.275
Margin	+0.060	+0.201
Mean	+0.10	+0.45

Table 1: Kendall’s tau correlation between human and each judgment using GEMINI-3.5-Flash. Halluc. = hallucination avoidance; Rel. corr. = relation correctness; Label acc. = label accuracy; Edge cross. = edge crossing; Hierarchy = visual hierarchy; Margin = proper margin. LVLM receives the rendered image and paper text; Code-judge receives raw HTML/SVG source and paper text.

fidelity-oriented questions such as hallucination and label accuracy, where source code exposes exact labels, component membership, and connector endpoints that are difficult to recover reliably from pixels alone. The smaller but still positive gains on visual hierarchy and margin indicate that source inspection also helps with layout-quality judgments, although these criteria remain more visually subjective. Table 5 reports the same per-criterion breakdown together with GPT-5.5 results.

3.2 End-to-End Generation

Table 2 evaluates whether current models (OpenAI, 2023; Gemini Team, 2023; Anthropic, 2024; Dubey et al., 2024; Yang et al., 2024; DeepSeek-AI, 2024; Jiang et al., 2023) can generate paper-grounded architectural figures as code. Besides direct paper-to-HTML generation, we include common strengthening strategies: four-phase intermediate planning (Wei et al., 2022; Zhou et al., 2023), retrieval-augmented generation (Lewis et al., 2020), ASCII intermediate diagrams, and supervised fine-tuning on FIGURECODE.

Specifically, the direct paper-to-HTML generation asks LLMs to write a self-contained HTML/SVG figure from the paper text. The planned setting forces the model to generate the four-phase intermediate plan used in our dataset before code generation. The RAG setting retrieves visually similar figures and supplies their source as references. The ASCII setting forces a coarse text diagram before HTML/SVG generation. Finally, we fine-tune a smaller open model, Qwen3.6-27B (Qwen Team, 2025, 2026), on FIGURECODE. Detailed experimental setups are in Appendices K,

Method	Fidelity			Layout			Overall
	Halluc.	Rel.	Label	Cross	Hier.	Margin	Avg.
(Ref) Human-verified GT Dataset GT HTML	4.10	4.07	4.67	4.43	4.17	4.13	4.26
(a) T2I baselines Nanobanana, paper-only	4.00	3.33	4.43	4.10	4.60	4.07	4.09
(b) Frontier LLM, direct paper-to-HTML GPT-5.5 (paper→HTML, 1 call)	4.10	4.73	4.93	4.17	4.57	4.60	4.52
(c) Open-source $\geq 100B$, planning DeepSeek-V4-Pro	4.17	3.67	4.57	3.93	4.07	4.43	4.14
(d) Frontier, planning GPT-5.5 (paper→self-CoT→HTML)	4.20	4.53	4.87	3.90	4.87	4.20	4.43
(e) Retrieval-augmented generation GPT-5.5 + image-RAG	4.07	4.70	4.90	4.10	4.90	4.23	4.48
(f) ASCII intermediate GPT-5.5 (paper→ASCII→HTML)	4.10	4.50	4.93	4.50	4.80	4.43	4.54
(g) Ours: supervised fine-tune on FigureCode SFT 4-LoRA cascade CoT → SFT-HTML qwen3.6-27B	1.10	1.17	1.17	4.43	3.70	4.53	2.68

Table 2: Main results on FIGURECODE. Overall denotes the average of the six representative sub-metrics. The GT row is scored with an 8-page paper context, whereas generated figures are scored with the methodology excerpt (Appendix E). Full results reported in Appendix Tab. 4.

L, and M.

The T2I baseline collapses on relation correctness and falls below the human-GT reference on every fidelity axis, while frontier code-generation pipelines stay within or above the GT band (Appendix E). ASCII intermediate diagrams and retrieval-augmented generation are the two most consistent strengthening strategies. Across all six axes the image-RAG mean and the ASCII mean exceed the strongest T2I baseline. The SFT results on the 27B model achieved low fidelity scores, likely due to its limited model capacity, but showed partial improvements in layout. These results indicate that LLMs need spatial sketches (ASCII), visually similar reference code (RAG), or fine-tuning to produce paper-grounded figures at GT-comparable quality.

3.3 Editability, Searchability, and Accessibility

We next evaluate properties that images alone struggle to support. For editability, we sample three structural edit instructions, apply each instruction to every figure, and verify both whether the requested change is made and whether unrelated structure is preserved. For searchability, we sample fifty-nine search instructions, retrieve figures using code and intermediate-plan text, and compare against baselines which retrieve figures us-

Property	Image-based	Code-based	Gain
Editability	54.3%	84.3%	30%
Searchability	27.4%	52.7%	25.3%
Accessibility	-	66.1%	66.1%

Table 3: Comparison of three functional properties between image-based and code-based figure generation. Editability is rate of edit success; searchability is F1@5 against the image-based baseline; accessibility is WCAG/ARIA coverage.

ing SigLIP-2 (Tschannen et al., 2025) image-text embedding similarities. The full instruction set is listed in Appendix N. For accessibility, we run browser-based checks whether each HTML/SVG source code covers WCAG/ARIA.

As shown in Tab. 3, code editing achieves a success rate of 84.3%, significantly outperforming the image-editing baseline at 54.3%. Retrieval also reaches an F1@5 of 52.7%, compared to 0.274 for SigLIP-2. Furthermore, the source code in our benchmark successfully complies with WCAG/ARIA standards at a rate of 66.1%.

4 Conclusion

We introduced FIGURECODE, a benchmark that re-frames scientific architectural figure generation as HTML/SVG source-code generation. Our results show that code-based evaluation better aligns with

human judgment and supports editability, searchability, and accessibility, while revealing that current LLMs still require spatial sketches, visually similar retrieved code, or fine-tuning to approach GT-comparable paper-grounded generation quality.

5 Limitations

The data collection for FIGURECODE is restricted exclusively to papers published in ACL proceedings between 2024 and 2025; it does not incorporate diverse visual and textual resources from other premier machine learning and computer vision venues, such as CVPR, NeurIPS, or ICML. Expanding the dataset to encompass a broader spectrum of research domains remains for future work.

Although our code-based evaluation framework significantly enhances alignment with human judgment compared to utilizing standalone Large Vision-Language Models (LVLMs), it still exhibits a moderate correlation coefficient ($\tau = 0.45$). Future research could explore more tightly correlated evaluation methodologies, such as hybrid assessment paradigms that combine both LVLM reasoning and code-based verification.

Acknowledgements

We thank anonymous reviewers for their constructive and insightful comments. K. Jung is with ASRI, Seoul National University, Korea. This work was supported by LG AI Research, and also partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2022-II220184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics).

References

- Anthropic. 2024. [The Claude 3 model family: Opus, sonnet, haiku](#). *Anthropic Model Card*.
- Jonas Belouadi, Anne Lauscher, and Steffen Eger. 2024. Automatiz: Text-guided synthesis of scientific vector graphics with tikz. In *International Conference on Learning Representations (ICLR)*.
- Tony Beltramelli. 2017. pix2code: Generating code from a graphical user interface screenshot. *arXiv preprint arXiv:1705.07962*.
- James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. 2023. [Improving image generation with better captions](#). *OpenAI Technical Report*.
- Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402.
- Siqi Chen, Xinyu Dong, Haolei Xu, Xingyu Wu, Fei Tang, Hang Zhang, Yuchen Yan, Linjuan Wu, Wenqi Zhang, Guiyang Hou, Yongliang Shen, Weiming Lu, and Yueting Zhuang. 2025. SVGenius: Benchmarking LLMs in SVG understanding, editing and generation. *arXiv preprint arXiv:2506.03139*.
- Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. In *Transactions on Machine Learning Research*.
- DeepSeek-AI. 2024. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. 2024. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*.
- Gemini Team. 2023. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.
- Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2023. Prompt-to-prompt image editing with cross-attention control. In *International Conference on Learning Representations*.
- Siyuan Huang, Yutong Gao, Juyang Bai, Yifan Zhou, Zi Yin, Xinxin Liu, Rama Chellappa, Chun Pong Lau, Sayan Nag, Cheng Peng, and Shraman Pramanick. 2026. Scifig: Towards automating scientific figure generation. *arXiv preprint arXiv:2601.04390*.
- Yerin Hwang, Dongryeol Lee, Kyungmin Min, Taegwan Kang, Yong-il Kim, and Kyomin Jung. 2025. Fooling the LVLM judges: Visual biases in LVLM-based evaluation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*.

- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.
- Hugo Laurençon, Léo Tronchon, and Victor Sanh. 2024. Unlocking the conversion of web screenshots into HTML code with the WebSight dataset. *arXiv preprint arXiv:2403.09029*.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*.
- Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The AI scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*.
- Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. 2022. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*.
- Tamara Munzner. 2014. *Visualization Analysis and Design*. CRC Press.
- OpenAI. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Wei Pang, Kevin Qinghong Lin, Xiangru Jian, Xi He, and Philip Torr. 2025. Paper2poster: Towards multimodal poster automation from scientific papers. *arXiv preprint arXiv:2505.21497*.
- Qwen Team. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Qwen Team. 2026. Qwen3.6-72B: Model card. <https://huggingface.co/Qwen/Qwen3.6-72B>. Accessed 2026-08-22.
- Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with CLIP latents. In *arXiv preprint arXiv:2204.06125*.
- Juan A. Rodriguez, Abhay Puri, Shubham Agarwal, Issam H. Laradji, Pau Rodriguez, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. 2025. Starvector: Generating scalable vector graphics code from images and text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. 2022. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*.
- Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. 2025. Agent laboratory: Using LLM agents as research assistants. *arXiv preprint arXiv:2501.04227*.
- Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. 2025. *Design2code: Benchmarking multimodal code generation for automated front-end engineering*. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3956–3974, Albuquerque, New Mexico. Association for Computational Linguistics.
- Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patherdhan. 2025. Paperbench: Evaluating AI’s ability to replicate AI research. *arXiv preprint arXiv:2504.01848*.
- Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. 2025. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*.
- Edward R. Tufte. 2001. *The Visual Display of Quantitative Information*, 2 edition. Graphics Press.
- Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, Alexis Chevalier, Sanjeev Arora, and Danqi Chen. 2024. Charxiv: Charting gaps in realistic chart understanding in multimodal LLMs. In *Advances in Neural Information Processing Systems*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.
- World Wide Web Consortium. 2018. *Scalable vector graphics (SVG) 2*. W3C Candidate Recommendation.
- Chengyue Wu, Zhixuan Liang, Yixiao Ge, Qiushan Guo, Zeyu Lu, Jiahao Wang, Ying Shan, and Ping Luo. 2025. *Plot2code: A comprehensive benchmark for evaluating multi-modal large language models*.

in code generation from scientific plots. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3006–3028, Albuquerque, New Mexico. Association for Computational Linguistics.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Cheng Yang, Chufan Shi, Yaxin Liu, Bo Shui, Junjie Wang, Mohan Jing, Linran Xu, Xinyu Zhu, Siheng Li, Yuxiang Zhang, Gongye Liu, Xiaomei Nie, Deng Cai, and Yujiu Yang. 2025. Chartmimic: Evaluating LMM’s cross-modal reasoning capability via chart-to-code generation. In *International Conference on Learning Representations*.

Xuanle Zhao, Xianzhen Luo, Qi Shi, Chi Chen, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2025. [Chart-coder: Advancing multimodal large language model for chart-to-code generation](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7333–7348, Vienna, Austria. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations*.

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

A Full Model Sweep

Table 4 reports the full model sweep across all generation settings, including the T2I baselines, direct HTML generation, planning-based generation, retrieval-augmented generation, ASCII-intermediate generation, and supervised fine-tuning variants.

B Human Evaluation Details

Recruitment and compensation. We recruited 10 Ph.D. students fluent in English who had

prior experience reading and evaluating machine-learning and NLP papers. The recruited annotators were provided with a detailed description of the task definition, the scoring rubric, anchor descriptions for each Likert level, and reference samples drawn from the training pool. All applicants were informed that their annotations would be used for academic purposes and would be published in paper material through the recruitment announcement and the per-task instructions. For the payment of the annotators, the co-authors first conducted annotations for several hours to estimate the average number of items that could be completed in the same time, and then converted that throughput estimate into an hourly rate at or above the local statutory minimum wage.

Task setup and aggregation. We sampled 60 architectural figures across recent ACL venue papers for the human-correlation study. Every annotator saw the rendered figure image and the full text of the corresponding paper, and responded to the six-question rubric described below. All scores are integer 1–5 Likert ratings. The 60 figures are split into two pools: a 20-figure *overlap subset* that is independently scored by all 10 annotators (used for inter-annotator agreement) and a 40-figure *single-annotator pool* in which each figure is scored by exactly one annotator, with the assignment balanced across annotators so that each annotator scores four figures from this pool. For every (figure, question) pair we reduce the annotator scores to a single human score by majority vote (mode); ties are broken toward the lower of the tied scores, a conservative choice that does not inflate human ratings. All Kendall’s τ numbers reported against human ratings use these per-(figure, question) mode-aggregated scores.

Rubric shown to annotators. The rubric used for human annotation mirrors the source-judge rubric (Appendix C) and consists of six questions organized under two axes:

- Fidelity: Hallucination; Relation Correctness; Label Accuracy.
- Layout: Edge Crossing; Visual Hierarchy; Proper Margin.

The exact questions presented to annotators are reproduced in Figure 2. The scoring scale is applied consistently across every question: 5 fully meets the criterion with no notable issues, 4 mostly meets

it with minor issues only, 3 partially meets it with one notable issue, 2 largely fails it with multiple notable issues, and 1 does not meet it at all.

Inter-annotator agreement. We measured inter-annotator agreement using Gwet’s AC2 with ordinal weights on the 20-figure overlap subset described above, in which all 10 annotators independently scored every figure on the six-question rubric. Across the six questions (Hallucination, Relation Correctness, Label Accuracy, Crossing, Hierarchy, and Margin), Gwet’s AC2 values were 0.89, 0.89, 0.89, 0.55, 0.30, and 0.43, with a mean of 0.66, indicating substantial agreement on the fidelity-oriented questions and moderate agreement on the layout-oriented questions. The agreement was strongest on Hallucination, Relation Correctness, and Label Accuracy, which is consistent with these being the most paper-grounded questions and therefore the most defensible target for source-grounded LLM judging.

Statistical analysis. For every Kendall’s τ reported against human ratings (Table 1 and Table 5) we additionally report a 95% confidence interval and test whether the source (code) judge correlates with humans strictly better than the raster (LVLM) judge. Confidence intervals are obtained by a non-parametric bootstrap that resamples the 60 evaluation figures with replacement 1,000 times; for each resample we recompute the per-(figure, question) mode-aggregated human score and the corresponding judge score and then recompute τ_b , taking the 2.5/97.5 empirical percentiles as the CI bounds.

C LLM-Judge Evaluation Details

Evidence-grounded source judge (primary). All headline numbers in Table 2 and the left of every slash in Table 4 use the evidence-grounded source judge with GPT-5.5, accessed through the OpenAI API. The right of every slash in Table 4 reports the same rubric under GEMINI-3.5-FLASH on all figures, as a sensitivity / cross-judge agreement column. Crucially, alongside each score the judge must emit compact HTML evidence snippets (quoted labels, SVG text, DOM class names, arrows, geometry or style fragments) and compact paper evidence snippets that justify the score; this evidence requirement is the source of the “evidence-grounded” label.

The full instruction issued to the judge is reproduced in Figure 3. The rubric block embedded in

the prompt lists the same six questions used in the human rubric in Appendix B.

The six rubric questions are Hallucination, Relation Correctness, Label Accuracy, Crossing, Hierarchy, and Margin — identical to the questions used in the human study. Each evaluation row is processed by four parallel shards and the per-row scores are merged.

LVLM judge (baseline). The LVLM judge is identical to the source judge except that the figure is supplied as a rendered PNG image instead of HTML source. Evidence snippets are also required. The same model and the same six questions are used, so the LVLM-versus-code comparison isolates the modality of the figure representation rather than the judge model or the rubric.

D Comparison with Related Tasks

Input–output contract. Every prior code-from-visual task supplies a rendered target at inference time. Screenshot-to-code (Beltramelli, 2017; Laurençon et al., 2024; Si et al., 2025) receives a target UI image and performs perceptual reconstruction, so the answer is fully specified by the pixels. FIGURECODE provides no target image: the input is paper text, and the system must decide which components, relations, and labels the figure should contain from the paper’s claims. This is a content-selection problem rather than an image-tracing one, and our judge scores exactly this paper-grounding.

Scientific figure generation. SciFig (Huang et al., 2026), PaperBanana (Zhu et al., 2026), and Paper2Poster (Pang et al., 2025) produce T2I pixels or constrained DSL outputs (e.g., draw.io). We are code-native and reuse the generated source as the evaluation substrate and for editing, retrieval, and accessibility (Tables 1 and 3).

Text-to-diagram in constrained DSLs. Mermaid- or TikZ-style generation such as AutomaTikZ (Belouadi et al., 2024) targets DSLs that cannot express the rich, nested, free-form elements of real paper figures, which motivates full HTML/SVG (Section 2.1).

Generic SVG generation. SVGGenius (Chen et al., 2025) and StarVector (Rodriguez et al., 2025) produce icons and free-form vector art that are not grounded in a source document.

Chart/plot-to-code. ChartMimic (Yang et al., 2025), Plot2Code (Wu et al., 2025), Chart-

- Q1. Hallucination.** Does the figure avoid including components, modules, or formulas that are not mentioned in the paper?
- Q2. Relation Correctness.** Do the arrows, connectors, and group containments faithfully reflect the input–output and component-to-component relations described in the paper?
- Q3. Label Accuracy.** Do the textual labels on modules, arrows, and annotations match the terminology used in the paper?
- Q4. Edge Crossing.** Are connectors and arrows routed so that crossings, overlaps, and visually ambiguous junctions are kept to a minimum?
- Q5. Visual Hierarchy.** Does the figure use grouping, sizing, and emphasis to convey which components are primary versus subsidiary?
- Q6. Proper Margin.** Are spacing, padding, and alignment between elements consistent so that no region feels crowded or arbitrarily empty?

Figure 2: The six human-evaluation questions presented to annotators. Each question is scored on a 1–5 Likert scale, applied consistently across all questions.

Role. You are an expert reviewer evaluating a scientific architectural figure against the paper it describes.

Input. The raw HTML/CSS/SVG source of the figure and a paper-text excerpt.

Evidence requirement. Base your judgment on explicit evidence in the HTML source and paper text; do not rely on imagined visual content. For each question, cite compact evidence snippets that justify the score:

- *HTML evidence* should quote labels, SVG text, DOM class names, arrows or paths, geometry or style snippets, or other raw-source fragments.
- *Paper evidence* should quote or paraphrase compact paper-text snippets that the HTML is being checked against.

Rubric. Evaluate each of the six questions (grouped under Fidelity and Layout; see Appendix B) on a 1–5 Likert scale, applied consistently across all questions.

Output. Return strict JSON only, with one top-level key per question. Each value is an object with fields “score” (integer 1–5), “reason” (a brief judgment tied to the cited evidence), “html_evidence” (a list of one to three compact HTML snippets), and “paper_evidence” (a list of compact paper-text snippets). Keep every evidence snippet under twenty words. Emit no commentary outside the JSON.

Figure 3: Evidence-grounded source-judge instruction. The judge sees the raw HTML/SVG source and a paper excerpt and must score each of the six rubric questions with HTML- and paper-grounded evidence snippets.

Coder (Zhao et al., 2025), and CharXiv (Wang et al., 2024) regenerate data-encoding charts from numeric series. We target architecture/method diagrams whose semantics are topology, containment, and inter-component relations, with no data channel.

In short, only our task combines (a) semantic generation from paper text without a target image, (b) architecture diagrams rather than charts or icons, and (c) a code representation used for evaluation itself.

E Analysis of Ground-Truth Deductions

To understand why some generated baselines score above the human-verified GT in Table 2, we categorized every fidelity deduction (score ≤ 4 on hallucination, relation, or label) that the GPT-5.5 judge issued over the 120 GT figures (231 deductions in total), using the judge’s written reason, into four categories (Table 6).

Category	Hal.	Rel.	Lab.	Total	%
(A) Decorative element	66	0	0	66	28.6
(B) Outside excerpt	9	3	0	12	5.2
(C) Genuine flaw	7	77	54	138	59.7
(D) Other / unclear	2	11	2	15	6.5
Total	84	91	56	231	100

Table 6: Taxonomy of the 231 fidelity deductions issued by the GPT-5.5 judge on the 120 GT figures: (A) human decorative element penalized, (B) content grounded in the paper but outside the judge’s excerpt, (C) genuine reconstruction flaw. On the hallucination question, 75 of 84 deductions (89%) are judge-side artifacts (A+B).

On the hallucination question, 75 of 84 deductions (89%) are judge-side artifacts: 66 penalize human decorative elements (icons, logos, UI mock-ups) and 9 penalize content grounded in the paper but outside the judge’s excerpt. A generated figure conditioned on the excerpt draws only what that text states and therefore cannot lose these points, which explains why conservative generators tie or beat GT on hallucination. The remaining

60% of deductions, concentrated on relation and label, are genuine reconstruction imperfections—predominantly minor 4-point deductions such as an ambiguous arrow direction or a typo—indicating that the judge discriminates real flaws rather than rewarding visual regularity. Consistently, the GT score rises from 3.80 to 4.26 (GPT-5.5) and from 4.57 to 4.70 (Gemini) when the judge’s context grows from the methodology excerpt to eight pages; the GT rows in Tables 2 and 4 use the eight-page context. We therefore report the GT row as a calibrated reference point, not an upper bound.

F Source-Verifiable Probe Study

To test whether some properties are verifiable only from source, we built balanced yes/no probes with mechanically decidable answers extracted from the DOM/SVG of GT figures, and posed each identical question to the same judge model (Gemini-3.5-Flash) under two conditions differing only in modality: the raw HTML source (Code) or the rendered PNG (LVLM). Because every probe has a known answer, we report accuracy and a McNemar test. Four property types are probed: (T1) exact small-font labels ($\leq 11\text{px}$), (T2) group membership expressed only through DOM containment with no visual cue, (T3) connector endpoints among crossing connectors, and (T4) text present in the source but clipped from the render. Because naturally clipped text is rare (1% of figures), T4 probes inject a short annotation into the source and clip it with `overflow:hidden`, verifying the clip in the render. Across the 200 evaluation figures, small labels occur in 44% and cue-free groups in 72%, whereas crossing connectors (2%) and clipped text (1%) are rare.

Property	<i>n</i>	Code	LVLM	<i>p</i>
(T1) Small label	60	0.98	0.98	1.00
(T2) DOM containment	60	0.82	0.70	0.09
(T3) Connector endpoint	48	0.60	0.65	0.73
(T4) Clipped text	60	1.00	0.62	$< 10^{-4}$

Table 7: Probe accuracy of the same judge model reading the source (Code) versus the rendered image (LVLM). *p*: McNemar test.

Content that exists in the source but is not painted into the pixels (T4) is verifiable from code and, by construction, invisible to any raster judge (1.00 vs. 0.62, $p < 10^{-4}$); the LVLM’s own evidence reads, e.g., “the text on the right is cut off.” Cue-free containment (T2) shows the same

direction more modestly. Small labels (T1) are read equally well in both modalities, and crossing-connector endpoints (T3) are hard for both. T4 is also the class of information the accessibility axis depends on, since screen-reader text lives only in the markup.

G Judge Robustness

Cross-judge agreement. Every generation track in Table 4 is scored by two independent judges, GPT-5.5 and Gemini-3.5-Flash. Across the 35 tracks, their overall scores agree at Spearman $\rho = 0.816$, and the code-over-image ordering holds under both.

Prompt and model variants. Table 8 scores the same 60 human-rated figures under four code-judge configurations and two LVLM references. Every code variant exceeds every LVLM variant. Requiring auditable evidence snippets trades some correlation ($0.452 \rightarrow 0.355$) but remains about $3\times$ the LVLM τ , and per-figure rankings of the two Gemini prompts stay closely aligned (Spearman 0.709).

Judge	Hal.	Rel.	Lab.	Cro.	Hie.	Mar.	Mean
Code/Gem. (orig.)	+0.766	+0.458	+0.715	+0.308	+0.275	+0.201	+0.454
Code/Gem. (evid.)	+0.528	+0.316	+0.477	+0.262	+0.312	+0.238	+0.355
Code/GPT-5.5 (evid.)	+0.246	+0.112	+0.286	+0.399	+0.061	+0.224	+0.221
Code/GPT-5.2	+0.166	+0.021	+0.213	+0.458	+0.216	+0.156	+0.205
LVLM/GPT-5.5	+0.127	+0.028	+0.164	+0.121	+0.345	+0.188	+0.162
LVLM/Gem.	+0.132	+0.083	+0.250	−0.121	+0.211	+0.060	+0.103

Table 8: Kendall’s τ against human ratings on the 60-figure set under judge prompt and model variants (Gem. = Gemini-3.5-Flash; evid. = evidence-required prompt). The first and last rows are the judges of Table 1.

H Direct Human Rating of Rendered Outputs

Because T2I outputs are reverse-converted to HTML before judging (Section 2.3), we checked whether this distorts the comparison by having two Ph.D. annotators rate the rendered pixels directly. We paired the T2I baseline (Nanobanana, paper-only) with the strongest non-T2I baseline (GPT-5.5 paper→ASCII→HTML, rendered to PNG) on 10 figures and asked the same six-question rubric on the images alone.

Axis	Raster		Code	
	Human	Judge	Human	Judge
Fidelity	4.83	4.83	4.75	4.70
Layout	3.87	4.87	4.30	4.60

Table 9: Mean 1–5 ratings of rendered outputs by humans versus our judge ($n = 10$ figures, 2 annotators). Raster = Nanobanana T2I; Code = GPT-5.5 ASCII→HTML rendered to PNG.

On fidelity, humans viewing the raw raster reach the same verdict as the code judge on the reverse-converted HTML (4.83 vs. 4.83; mean per-item $|\text{human} - \text{judge}| = 0.33$), so reverse conversion does not distort the fidelity comparison. Humans and the judge also agree on the ranking (both place the raster slightly above the code baseline overall; human $\Delta = -0.31$, n.s.; judge -0.20) and on the direction of five of six questions. On layout, the judge sits near ceiling while humans discriminate more; the two annotators themselves agree far less on layout (exact/within-1 agreement 0.32/0.58) than on fidelity (0.72/0.93), reproducing the axis split of Appendix B.

I Detailed Instructions for Generation

We summarize the exact instructions issued to every generation track in the model sweep. Per-track user messages differ only in which fields (paper body text, reference images, ASCII intermediate, or prior CoT phases) are inserted into the template; the system instructions described below are reproduced from the production code.

Direct paper-to-HTML. The model is positioned as an expert scientific figure reconstruction model and a senior HTML/CSS/SVG developer. The verbatim system instruction is reproduced in Figure 4. The user message simply contains the paper body and a directive to output only HTML.

Four-phase CoT cascade. The CoT track decomposes figure design into four specialist phases: topology (Phase 1), layout (Phase 2), styling (Phase 3), and structural intermediate representation (Phase 4). Each phase has its own system instruction; downstream phases receive the prior phases as additional context but are forbidden from re-emitting them. The four phase instructions are reproduced in Figures 5–8. The final HTML synthesizer receives the concatenated four-phase plan plus the paper body and is asked to emit a single

self-contained HTML document under the same hard constraints as the direct prompt.

Image-RAG with visual calibration. For image-RAG, the user message additionally contains two reference figure images retrieved from the training pool (encoded as inline image parts). The calibration directive shown in the user message is reproduced in Figure 9. The system instruction otherwise reuses the direct-HTML constraints described above.

ASCII intermediate. The ASCII track is two-stage. The Stage 1 instruction turns the paper body into a terminal-readable diagram (Figure 10). In Stage 2 the direct-HTML system instruction is reused with one added constraint, namely that the HTML must preserve the ASCII diagram’s structure, ordering, labels, arrows, grouping, and side notes. The Stage 2 user message contains both the paper body and the ASCII diagram from Stage 1.

J Detailed Instructions for Dataset Generation

Source figures. The FIGURECODE evaluation set is a stratified sample drawn from a 1,369-paper architectural-figure corpus from recent ACL Anthology. For each paper we selected the single main architectural or methodology figure, excluding result plots and example tables.

Reverse HTML/SVG reconstruction (ground truth). Each raster figure is paired with a hand-curated HTML/SVG source that visually reproduces the original layout. The reconstruction prompt asks an HTML/CSS expert model to produce a single self-contained file using only HTML, CSS, and SVG, and to preserve the original figure’s group structure, arrow topology, label text, and visual hierarchy. The reconstructor sees both the raster figure and the paper’s text, so labels match the paper’s terminology rather than the often-illegible PNG. After model generation, each candidate HTML is re-rendered to PNG and compared with the original by the same human annotators on a 1–5 fidelity check; only figures with reconstruction score at least 4 enter the dataset.

Golden four-phase CoT annotation. For every training and evaluation figure we additionally distill a “golden CoT” by prompting a frontier model (GPT-5.5) with both the paper body and the ground-truth HTML, and asking it to emit four separate

Role. You are an expert scientific figure reconstruction model and a senior HTML/CSS/SVG developer.

Goal. Given a research paper body, generate one self-contained HTML document that reconstructs the likely methodology-style figure for the paper.

Hard constraints.

- Output only valid HTML; no markdown fences, explanations, headings, analysis, or thinking tags.
- Use only HTML, CSS, and SVG. No canvas, no JavaScript, no external libraries, no external fonts, no network access.
- Build a complete scientific diagram: title, major modules, grouping containers, arrows or connectors, labels, and visual hierarchy.
- Text must be visible and readable. Minimum font size: 12 px for normal labels, 10 px only for short secondary annotations. Do not hide essential text with overflow-hidden; wrap long labels onto 2–3 lines or shorten them.
- Prefer a stable 16:9 or 4:3 root figure with explicit width and height or aspect ratio.
- Always close the style, head, body, and html tags.
- The first non-whitespace characters must be a doctype declaration or an opening html tag.

Figure 4: Direct paper-to-HTML generation instruction. The model receives the paper body and must emit a single self-contained HTML/CSS/SVG document for the paper’s main methodology figure.

phase blocks following the cascade specification in Appendix I. The Phase 1 through Phase 4 prompts shown to the annotator model are identical to the production cascade prompts above except that they include the line “A reference HTML rendering of the final figure is provided as additional grounding; do not copy its tags, only use it to anchor your design choices.” Each generated CoT is then run through a self-consistency check: the same Phase-4 IR is re-rendered into HTML by a separate model, and the resulting figure must score at least 4 on the same human fidelity criterion. CoTs that fail are discarded and re-generated up to three times. The released dataset schema therefore stores, per paper, the fields paper text, ground-truth HTML, and a CoT dictionary with four entries (topology, layout, styling, and structural IR).

K SFT Setup

We fine-tune two families of student models on the FigureCode training pool. Both setups run on a single NVIDIA A100 GPU.

Phase-specialist four-LoRA cascade. Each phase from 1 through 4 trains an independent LoRA adapter that consumes the paper text plus the prior frozen golden phases, and produces only the target phase block. The base model is Qwen3.6-27B, loaded with Unsloth in 4-bit quantization and bfloat16 dtype. The LoRA configuration uses rank 16, alpha 32, zero dropout, target modules consisting of the query, key, value,

and output projection matrices, no bias adaptation, gradient checkpointing. At inference time the four LoRA adapters are stitched together: phase k sees the concatenation of all earlier phases and emits its own block, and the final Phase-4 IR is then fed to the HTML synthesizer.

End-to-end CoT LoRA. The same base model is fine-tuned with a deliberately shallow LoRA so that vanilla performance is preserved: rank 8, alpha 8, zero dropout, and target modules restricted to the query and value projections only. Optimizer settings are otherwise identical to the cascade above; training runs for one epoch.

L RAG Setup

Retrieval pool. The retrieval pool is a deduplicated union of two training shards, keeping only entries whose golden topology block has at least eighty characters and de-duplicating by SHA1 of the paper text. The final pool has 1,000 entries and is cached as a pickled list of entry records together with a pre-computed embedding tensor.

Embedding model. We use BAAI BGE-base-en, version 1.5, a 768-dimensional encoder, for both the query and document sides. Both the pool documents and the query topology field are encoded with CLS pooling, truncated at 512 tokens, batch size 8, and then L2-normalized. Pool embeddings are pre-computed once and reused across all retrieval calls.

Role. You are an expert technical illustrator and UI/UX developer. You are given the body text of a research paper.
Goal. Produce only the topology section — the topological skeleton of the figure that downstream layout, styling, and structural-IR phases will build on.

Critical rules.

- Write as if you are conceptualizing the diagram from scratch based purely on reading the paper.
- Invent and specify highly exact values: pixel sizes, hex color codes, flex or grid directions, gaps, paddings, semantic class names. The downstream model relies on your confident, precise numerical assignments.
- Do not include any code blocks or HTML; only reasoning text.

Output discipline.

- Tag each factual or design claim as Paper-grounded, Design inference, or Implementation choice.
- When giving exact spatial values, present them in a two-layer form: first the constraint-level reasoning, then the exact implementation values.

Required content (in order). Module families; primary flow; secondary flow; merge or split structure; grouping principle; emphasis point.

Format. Output only one topology block, with no preamble or commentary.

Figure 5: Phase 1 (topology) instruction of the four-phase CoT cascade. Phases 2–4 share the same template and discipline rules but target the layout, styling, and structural-IR blocks respectively.

Retrieval. For each evaluation row we compute the cosine similarity of the query embedding against all 1,000 pool embeddings and keep the top two records. Each retrieved record is written to a per-row JSON file with the same schema as the original pilot. The image-RAG generation step reads this file on the fly and attaches the two reference figure images plus their golden topology snippets to the user message.

Searchability baselines. For the searchability experiment in Table 3 we additionally implemented two retrieval baselines. The caption-BM25 baseline indexes only the paper-provided figure captions, and the SigLIP-2 image-text baseline scores raw figure images against the natural-language query. The BM25 implementation uses the standard rank-bm25 library with whitespace tokenization on lowercased text. Our code-plus-CoT BM25 indexes the concatenation of the golden HTML’s visible text tokens with the four CoT phases. All three retrievers are scored on the same fifty-nine motif-grounded queries across figures.

M ASCII Intermediate Setup

Pipeline. The ASCII intermediate baseline uses a two-stage text bridge. Stage 1 receives only the paper body and emits a compact terminal-readable methodology diagram. Stage 2 receives the same paper body plus the generated ASCII diagram and converts it into a self-contained HTML/CSS/SVG

figure. No retrieval examples, ground-truth HTML, golden CoT, or target figure image are provided for the reported paper-to-ASCII-to-HTML rows.

Implementation and decoding. We use prompts to generate ASCII intermediate outputs detailed in 11.

N Editability and Searchability Instructions

Editability instructions. The editability experiment uses three structural edit instructions, each applied to every figure in the evaluation set for both the HTML/SVG editing condition and the raster image-editing condition. Figure 12 shows the three instructions in the exact form used by the evaluator-facing task, with only formatting shortened for space.

Searchability instructions. The searchability experiment uses fifty-nine motif-grounded retrieval instructions over the same evaluation set. The full set spans ten motif families: RAG variants, agent memory, multi-hop reasoning, vision-language pipelines, LLM-as-judge, alignment/preference, non-autoregressive generation, self-refinement loops, knowledge editing, and tool-augmented reasoning. To keep the paper appendix readable, Figure 13 reports representative instructions sampled across these families; all results in Table 3 are computed over the complete 59-instruction set.

Role. You are an expert technical illustrator and UI/UX developer. You are given the body text of a research paper and a partially completed figure-design CoT containing the Phase 1 topology block.

Goal. Produce only the layout section — the spatial specification that downstream styling and structural-IR phases will build on.

Critical rules.

- Write as if you are conceptualizing the diagram from scratch based purely on reading the paper and the provided topology.
- Invent and specify highly exact values: pixel sizes, hex color codes, flex or grid directions, gaps, paddings, and semantic class names. The downstream model relies on confident, precise numerical assignments.
- Do not include code blocks or HTML; only reasoning text.

Output discipline.

- Tag each factual or design claim as Paper-grounded, Design inference, or Implementation choice.
- When giving exact spatial values, present them in a two-layer form: first the constraint-level reasoning, then the exact implementation values.

Required content (in order). Canvas direction; macro hierarchy; branch arrangement; alignment law; spacing law; layout mechanism, including the concrete CSS engine choice and exact implementation values.

Format. Output only one `<phase2_layout>...</phase2_layout>` block. Do not re-emit Phase 1.

Figure 6: Phase 2 (layout) instruction of the four-phase CoT cascade.

O Qualitative Analysis

To complement the quantitative source-judge scores in Tab. 4, Figure 14 compares the outputs of every major generation track on a single representative paper (*Memory OS of AI Agent*, evaluation index 70). We pick this paper because (i) it is present in both the T2I rendering set and our HTML evaluation set, and (ii) the ground-truth figure is a typical scientific architecture diagram with multiple labeled modules, a hierarchical containment structure, and a directed data-flow with several feedback paths. Below we describe the recurring patterns we observed across the full sweep, using Figure 14 as the running example.

T2I raster baselines (panels b, c). The two raster generators reproduce the overall visual genre of an architecture figure (boxes, arrows, icons, and in panel b even the correct STM/MTM/LPM labels), and panel b is one of the visually most faithful outputs in the row. The cost of this fidelity is that the figure is a frozen pixel grid: small annotations such as “LPM persona vault”, “heat-based replacement”, or the formula $\text{Heat} = \alpha N + \beta L + \gamma R$ are either omitted or rendered at sub-pixel resolution that no downstream agent can parse. GPT-Image-2 (panel c) chooses the opposite trade-off: it crams in many sub-labels, but most are illegible at column width. Across the full sweep this is the dominant failure mode for T2I: the rendered image looks plausible at a glance, but cannot be verified,

edited, or re-used by a text-native agent.

Vanilla HTML generators (panels d, f, g). The two open-source 100B-class vanillas and the GEMINI-3.5-Flash baseline tend toward a sparse, flowchart-like layout that emphasizes a single linear flow at the expense of containment hierarchy. GEMINI-3.5-Flash (panel d) reduces the three-tier memory store to a single “Memory Storage” box with text bullets, dropping the explicit nesting of STM/MTM/LPM. DeepSeek V4-Pro (panel f) renders only the right half of its canvas at column width and leaves the rest as dark empty space, indicating either a width-estimation failure or an overly-large outer container; this layout-overflow pattern recurs in a non-trivial fraction of the DeepSeek outputs. Kimi K2.6 (panel g) is the strongest of this group: it preserves all three memory tiers and shows distinct Memory Update / Storage / Retrieval / Response panels, but uses thin, faint connector arrows that often cross.

Frontier HTML with planning, retrieval, or ASCII (panels e, h, i, j). GPT-5.5 in direct mode (panel e) already produces a four-stage layout with clear module headers, but its inline arrow routing is the weakest aspect; several arrows leave the right margin and re-enter from the top. Adding either the four-phase CoT plan (panel h), an image-RAG reference (panel i), or an ASCII intermediate (panel j) consistently improves the figure in the same direction: tighter grouping of related boxes, more

Role. You are an expert technical illustrator and UI/UX developer. You are given the body text of a research paper and a partially completed figure-design CoT containing the Phase 1 topology and Phase 2 layout blocks.

Goal. Produce only the styling section — the visual styling specification that the downstream structural-IR phase will reference.

Critical rules.

- Write as if you are conceptualizing the diagram from scratch based purely on reading the paper and the provided earlier phases.
- Invent and specify highly exact values: pixel sizes, hex color codes, flex or grid directions, gaps, paddings, and semantic class names. The downstream model relies on confident, precise numerical assignments.
- Do not include code blocks or HTML; only reasoning text.

Output discipline.

- Tag each factual or design claim as Paper-grounded, Design inference, or Implementation choice.
- When giving exact visual values, present them in a two-layer form: first the constraint-level reasoning, then the exact implementation values.

Required content (in order). Size families; semantic color mapping with exact hex codes; border system; typography hierarchy; connector style; highlight token for the emphasis point from Phase 1.

Format. Output only one `<phase3_styling>...</phase3_styling>` block. Do not re-emit Phases 1–2.

Figure 7: Phase 3 (styling) instruction of the four-phase CoT cascade.

explicit visual encodings (color-coded module families, dashed lines for update paths, solid lines for retrieval), and the appearance of paper-specific text such as the heat-based replacement formula or the FIFO eviction note. Of the three augmentations, ASCII most reliably yields a clean hierarchical layout, while image-RAG most reliably mimics the source paper’s panel arrangement.

Golden CoT upper bound (panel k). When the ground-truth four-phase plan is supplied as input, GPT-5.5 reaches a publication-quality figure with numbered modules, an explicit “closed memory loop” annotation, and a legend for line semantics. The remaining gap to the original paper figure is mainly stylistic — iconography, color choice, and the conversation-stream illustration on the left of panel (a). This gap quantifies how much of the residual error in the sweep comes from *plan generation* rather than from *HTML synthesis*: even our strongest synthesizer hits a low ceiling without a good plan, and self-generated planning remains the bottleneck once the plan quality is fixed.

SFT 4-LoRA cascade (panel l, Ours). Our 27B SFT cascade — four phase-specific LoRA adapters followed by an SFT-HTML synthesizer with no access to the ground-truth plan — recovers the full STM/MTM/LPM tier hierarchy with the correct nesting depth, names each tier, and additionally renders the two update operators (“STM-MTM Update / FIFO” and “MTM-LPM Update / Heat-

based thresholding”) as separate captioned modules. Compared with the open-source 100B-class vanillas (panels f, g) this is a comparable performance at roughly 1/4 the parameter count.

P Use of ChatGPT and Compliance with OpenAI’s Terms

We utilized OpenAI’s ChatGPT for limited assistance in refining the writing and formatting of this paper. All substantive contributions, including the core methodology, dataset construction, experiments, and analysis, were conducted independently by the authors. Our usage complies with OpenAI’s Terms of Use and Usage Policies.

Q Artifact licenses, terms of use, and intended use.

FIGURECODE is constructed from publicly available ACL Anthology papers and their architectural or methodology figures, which we use for research purposes only. We use the source papers and figures as scholarly artifacts for benchmark construction, evaluation, and analysis, and we do not redistribute the original papers as a replacement for the official ACL Anthology versions. The HTML/SVG reconstructions and intermediate plans released as part of FIGURECODE are intended for non-commercial research on scientific figure generation, code-based evaluation, editability, searchability, and accessibility. They should

Role. You are an expert technical illustrator and UI/UX developer. You are given the body text of a research paper and a partially completed figure-design CoT containing the Phase 1 topology, Phase 2 layout, and Phase 3 styling blocks.

Goal. Produce only the structural intermediate representation — the schema-dense ground truth that the downstream HTML generator will follow.

Critical rules.

- Enumerate every group, container, node, and edge as a separate entry, derived from the paper and the prior phases.
- Invent and specify highly exact values: pixel sizes, hex color codes, flex or grid directions, gaps, paddings, and semantic class names. The downstream model relies on confident, precise numerical assignments.
- Do not include code blocks or HTML; only reasoning text.

Output discipline.

- Tag each factual or design claim as Paper-grounded, Design inference, or Implementation choice.
- Use stable IR identifiers: g1, g2, ... for groups; sg1, sg2, ... for subgroups; n1, n2, ... for nodes; and e1, e2, ... for edges.

Required content (in order). Groups with id, parent, role, order, and data attributes; rendered subgroups with id, parent, role, border style, and emphasis; nodes with id, parent, type, role, and paper-implied data attributes; edges with id, source, target, kind, and routing notes; containment; internal arrangement; alignment rules; global layout; label placement; panel structure; rendered exemplars; connector families; abstraction policy.

Format. Output only one `<phase4_structural_ir>...</phase4_structural_ir>` block. Do not re-emit Phases 1–3.

Figure 8: Phase 4 (structural intermediate representation) instruction of the four-phase CoT cascade.

not be used to impersonate the original authors, to reproduce copyrighted paper content outside fair academic use, or to generate misleading scientific artifacts.

For external artifacts, we follow the corresponding access conditions and terms of use. Commercial and hosted model APIs, including OpenAI and Gemini models, are used only through their official interfaces and under their respective service terms. Open-source models, embedding models, and software packages are used under their public licenses or distribution terms. These include the LLMs and encoders used for generation, judging, fine-tuning, and retrieval, as well as standard libraries used for BM25 retrieval, rendering, and evaluation. We use these artifacts for the purposes for which they are made available, namely research experimentation, model evaluation, retrieval, and code generation. Any released code, data schema, and derived annotations will be distributed with explicit license information and documentation describing the intended research use of the benchmark.

R Privacy and content safety.

The benchmark is derived from publicly available scientific papers rather than private user data. We do not collect personal communications, social-media content, demographic profiles, or other private text. During curation, we focus on architectural and methodology figures and filter out

result plots, tables, and trivial text-box figures. The released benchmark is anonymized with respect to the submitting authors of this paper. Because the source papers are public scholarly documents, author names and citation information may appear in the original papers; however, the benchmark is intended to center on figure structure, paper-grounded labels, and HTML/SVG representations rather than personal information. We do not intentionally include offensive content, and the data domain—scientific architecture diagrams from ACL-venue papers—makes such content unlikely. Users should nevertheless treat the benchmark as a research artifact derived from public papers and should follow the citation, license, and use restrictions of the original sources.

Context. Below is the target paper body text and reference figures from related papers.
How to use the references. Use the reference figures only as visual or structural calibration:

- layout density and overall composition,
- shape vocabulary (rectangles, rounded boxes, lanes, panels),
- grouping style and container nesting,
- arrow style (solid / dashed / curved / labeled),
- label treatment (placement, size, hierarchy).

Hard restriction. Do not copy the references' scientific content, paper-specific terms, or exact labels. The final figure content must be derived from the target paper text alone.

Goal. Generate a single self-contained HTML file for the target paper's likely main methodology figure. Reference figures follow at the end of this user message.

Figure 9: Image-RAG calibration directive. Two reference figure images are attached to the user message; the model uses them only for visual/structural calibration, and all factual content must come from the target paper text.

Role. You are an expert scientific diagram planner.

Goal. Given a research paper body, produce a compact CLI/ASCII-style methodology diagram that a human can understand in a terminal.

Hard constraints.

- Output only the text diagram and short inline labels. No markdown fences, no prose explanation, no HTML, no thinking tags.
- Use ASCII or Unicode box-drawing sparingly: boxes, arrows, grouping brackets, lanes, and short labels.
- Capture the likely main methodology figure: stages, inputs, outputs, data flow, feedback or auxiliary paths, and key annotations.
- Keep text labels concise but explicit; avoid empty boxes.
- Keep the layout implementable as an HTML figure later, and prefer a layout that can map to 16:9 or 4:3 HTML.

Figure 10: ASCII intermediate, Stage 1 instruction. The model converts the paper body into a compact terminal-readable diagram; Stage 2 then renders this ASCII diagram, together with the paper body, as a polished HTML figure.

Stage 1. Given a research paper body, produce a compact CLI/ASCII-style methodology diagram that a human can understand in a terminal. Output only the text diagram and short inline labels; use boxes, arrows, grouping brackets, lanes, and concise labels; capture stages, inputs, outputs, data flow, feedback or auxiliary paths, and key annotations; keep the layout implementable as a 16:9 or 4:3 HTML figure.

Stage 2. Given the paper body and the intermediate ASCII/CLI diagram, generate one self-contained HTML document using only HTML, CSS, and SVG. Preserve the ASCII diagram's structure, ordering, labels, arrows, grouping, and side notes; keep all text visible and readable; avoid external libraries, JavaScript, canvas, external fonts, and network access; output only valid HTML.

Figure 11: ASCII intermediate instructions. Stage 1 creates the spatial sketch; Stage 2 renders the sketch and paper content into HTML/SVG.

Method	Fidelity			Layout		
	Halluc.	Rel.	Label	Cross	Hier.	Margin
<i>(Ref) Human-authored GT</i>						
Dataset GT HTML	4.10/4.73	4.07/4.73	4.67/4.73	4.43/4.70	4.17/4.73	4.13/4.73
<i>T2I baselines</i>						
Nanobanana, paper-only	4.00/4.90	3.33/4.86	4.43/4.91	4.10/4.88	4.60/4.93	4.07/4.95
Nanobanana, + 4-phase CoT plan	3.17/4.59	3.17/4.68	3.87/4.71	4.20/4.71	4.30/4.76	4.00/4.76
GPT-Image-2, paper-only	3.77/4.89	3.63/4.91	4.50/4.91	4.27/4.91	4.83/4.91	4.30/4.88
GPT-Image-2, + 4-phase CoT plan	3.53/4.74	3.67/4.81	4.67/4.81	4.30/4.76	4.57/4.82	4.10/4.82
<i>Frontier LLM, direct paper input</i>						
GPT-5.5 (paper→HTML, 1 call)	4.10/4.89	4.73/4.91	4.93/4.92	4.17/4.82	4.57/4.91	4.60/4.92
GEMINI 3.5 Flash (paper→HTML, 1 call)	4.27/4.74	3.60/4.39	4.67/4.82	4.20/4.70	3.73/4.57	4.50/4.79
<i>Open-source $\geq 100B$, end-to-end</i>						
Kimi K2.6	3.57/4.59	3.23/4.78	4.27/4.89	3.90/4.82	4.17/4.92	3.77/4.90
Kimi K2.5	3.33/4.59	3.07/4.79	4.17/4.85	3.97/4.79	4.13/4.92	3.87/4.88
Qwen3-Coder-480B	3.47/4.45	2.90/4.13	4.10/4.75	4.33/4.86	3.37/4.62	3.70/4.87
Qwen3-Coder-Next	3.30/4.30	2.70/4.15	3.87/4.44	3.57/4.23	3.70/4.43	3.37/4.30
Qwen2.35B-2507	3.70/3.69	2.60/3.53	3.43/3.71	3.33/4.16	3.23/3.82	2.53/4.01
DeepSeek-V4-Pro	4.17/4.81	3.67/4.89	4.57/4.94	3.93/4.87	4.07/4.91	4.43/4.95
DeepSeek-V3.2	3.87/4.57	3.27/4.60	4.53/4.77	3.83/4.61	4.00/4.86	3.57/4.77
GLM-5.1	4.30/4.64	3.87/4.60	4.17/4.61	4.53/4.64	3.93/4.60	3.87/4.61
GPT-OSS-120B	3.77/4.55	2.93/4.49	3.97/4.83	3.20/4.54	3.33/4.64	3.67/4.86
Llama-4 Maverick	4.20/4.90	2.40/3.39	4.20/4.87	4.50/4.79	3.20/4.00	3.87/4.70
Nemotron-3 Super-120B	3.40/4.16	2.33/3.51	3.80/4.08	3.60/4.39	2.97/4.02	3.03/3.99
ERNIE 4.5-300B	2.30/3.77	2.47/4.19	3.70/4.48	4.10/4.80	3.83/4.74	3.63/4.75
Mistral-Medium-3.5	3.73/4.68	3.00/4.39	4.30/4.86	3.83/4.71	3.77/4.81	3.77/4.75
<i>(d) Frontier, end-to-end</i>						
GPT-5.5 (paper→self-CoT→HTML)	4.20/4.86	4.53/4.90	4.87/4.90	3.90/4.83	4.87/4.90	4.20/4.90
GEMINI 3.5 Flash (paper→self-CoT→HTML)	4.20/4.82	3.63/4.86	4.53/4.95	3.13/4.54	4.10/4.92	4.00/4.88
<i>(e) Frontier + RAG ($k=2$)</i>						
GPT-5.5 + image-RAG	4.07/4.93	4.70/4.96	4.90/4.96	4.10/4.91	4.90/4.95	4.23/4.96
GEMINI 3.5 Flash + image-RAG	3.77/4.44	3.60/4.04	4.17/4.22	4.10/4.37	4.20/4.17	4.57/4.24
GPT-5.5 + HTML-RAG (text)	3.43/4.61	3.77/4.67	4.13/4.69	4.33/4.83	4.37/4.80	4.20/4.86
<i>(f) Frontier + ASCII intermediate</i>						
GPT-5.5 (paper→ASCII→HTML)	4.10/4.91	4.50/4.93	4.93/4.93	4.50/4.92	4.80/4.93	4.43/4.92
GEMINI 3.5 Flash (paper→ASCII→HTML)	4.07/4.84	4.03/4.84	4.53/4.86	4.83/4.86	4.50/4.83	4.83/4.86
<i>(g) Ours: SFT on FigureCode</i>						
SFT 4-LoRA cascade CoT → Qwen3.6-27B HTML	1.10/4.33	1.17/4.13	1.17/4.43	4.43/4.50	3.70/4.00	4.53/4.53
<i>(h) Golden CoT → HTML</i>						
Golden CoT + Vanilla qwen3.6-122B-A10B	3.00/4.67	3.25/4.33	4.08/4.42	4.33/4.92	3.92/4.75	4.50/4.92
Golden CoT + SFT Qwen3.6-27B	1.13/4.10	1.13/4.07	1.13/4.17	3.67/4.00	3.47/3.83	3.83/4.33
Golden CoT + GEMINI 3.5 Flash	3.10/4.53	3.00/4.42	3.80/4.59	3.73/4.79	3.67/4.67	4.00/4.82
Golden CoT + GPT-5.5	3.33/4.72	3.73/4.70	4.20/4.79	4.53/4.90	4.47/4.86	4.40/4.91

Table 4: Full model sweep on FIGURECODE with cross-judge sensitivity. Left of the dash: GPT-5.5’s judgment; right: GEMINI-3.5-FLASH’s. Both judges score raw HTML + the paper’s methodology section under the same six-question rubric. The *(Ref)* row is the dataset GT HTML scored by both judges with 8-page paper context (the methodology-only setting underestimates GT because the GT figure intentionally abstracts content from sections beyond methodology; the 8-page condition is the smallest paper context under which both judges stably evaluate GT). T2I rows are pre-processed through GEMINI-3.5-FLASH reverse-to-HTML. Group (h) isolates the HTML-synthesis sub-task by providing the ground-truth 4-phase plan as input. Across the 35 tracks the two judges agree at Spearman $\rho = 0.816$ on the overall score.

Metric	GPT-5.5 Image	GPT-5.5 Code	GEMINI Image	GEMINI Code
Hallucination	–	+0.166	+0.132	+0.766
Relation correctness	–	+0.021	+0.083	+0.458
Label accuracy	–	+0.213	+0.250	+0.715
Edge crossing	+0.250	+0.458	-0.121	+0.308
Visual hierarchy	+0.041	+0.202	+0.211	+0.275
Proper margin	+0.122	+0.196	+0.060	+0.201
Mean over six	–	+0.209	+0.103	+0.454

Table 5: Per-question Kendall’s τ against human ratings on the 60-figure human-correlation set used in Table 1. LVLM judges receive the rendered image and paper text; source judges receive raw HTML/SVG source and paper text. Dashes indicate undefined Kendall’s τ because GPT-5.5 Image scores were constant on the corresponding question.

Edge add. Add one new red arrow connecting the leftmost visible box/module to the rightmost visible box/module. Use an SVG line or path with an arrowhead, stroke #dc2626, stroke width 2px, and a small label “feedback” near the middle. Do not modify existing arrows.

Node insert. Insert one rectangular box labeled “Verifier” immediately above the largest existing box. Use text/border color #1e3a8a, fill #dbeafe, 1.5px border, and 6px rounded corners. Keep a small vertical gap and do not modify existing elements.

Module swap. Identify the two leftmost sibling modules at the same vertical level and swap their horizontal positions. Preserve their internal contents, sizes, and styles. Existing arrows should still point to the same semantic modules after the swap.

Figure 12: Editability instructions. Each instruction tests a different source-native operation: adding a relation, inserting a node, and rearranging sibling modules under preservation constraints.

RAG variants. Look for diagrams showing multi-hop retrieval where a complex query is decomposed into sequential sub-questions, with context retrieved iteratively for each step.

Agent memory. Look for diagrams showing a memory system that extracts entity-centric attributes and values from raw conversation history to build a structured semantic memory.

Multi-hop reasoning. Look for diagrams showing a tree-search reasoning framework where step-by-step reasoning paths are generated and evaluated by a process reward model.

Vision-language pipeline. Look for diagrams showing multimodal dense retrieval, where text queries and multimodal documents are encoded into a shared vector space.

LLM-as-judge. Look for diagrams showing an iterative generator–critic loop where a judge model evaluates an output and feeds critique back for refinement.

Alignment/preference. Look for diagrams showing a preference-learning setup contrasting chosen and rejected responses against a reference model.

Tool-augmented reasoning. Look for diagrams showing an agent loop of Thought, Action, and Observation between an LLM and an external environment or tool.

Figure 13: Representative searchability instructions. The complete evaluation contains fifty-nine motif-grounded instructions across the same motif families; this figure shows a compact sample rather than the full query list.

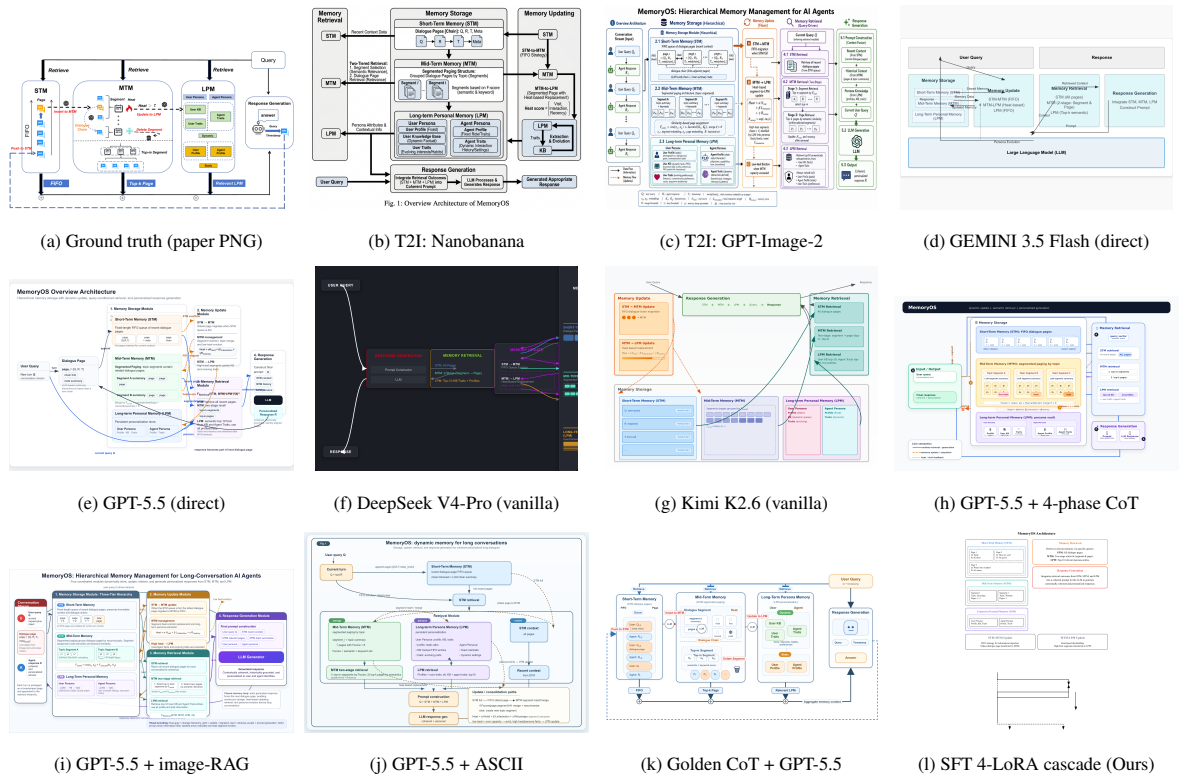


Figure 14: Qualitative comparison of every major track on *Memory OS of AI Agent* (index 70). Panel (a) is the original paper figure; (b)–(c) are raw T2I rasters; (d)–(g) are vanilla HTML generations from frontier and open-source models; (h)–(j) add a planning, retrieval, or ASCII intermediate step on top of GPT-5.5; (k) feeds the ground-truth four-phase plan to GPT-5.5 as an upper bound; (l) is our SFT 4-LoRA cascade (self-generated four-phase plan, then SFT-HTML qwen3.6-27B).