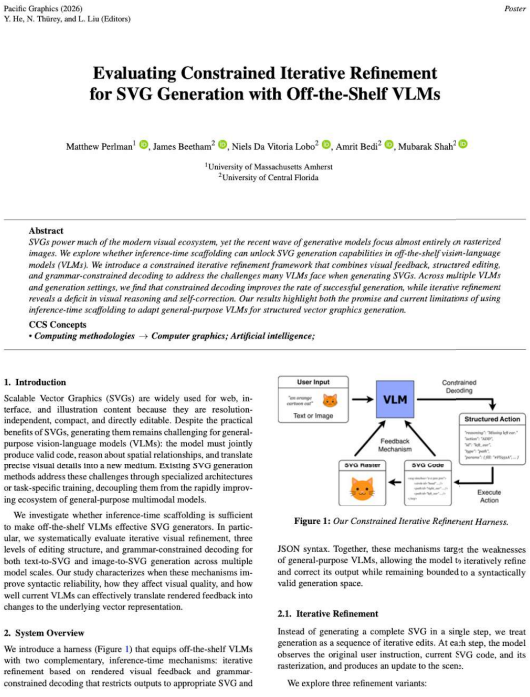
Week 8
Structured Vector Graphics Editing With VLMs

Matthew Perlman

James Beetham, Dr. Amrit Bedi, Dr. Mubarak Shah

Progress This Week

- Finished Writing Paper
- Qualitative GPT Examples
- Finalized System Diagram
- Compiled Table Results
- Official Pacific Graphics Submission



Pacific Graphics (2026)
Y. He, N. Thürey, and L. Liu (Editors)

Poster

Evaluating Constrained Iterative Refinement for SVG Generation with Off-the-Shelf VLMs

Matthew Perlman[1], James Beetham[2], Niels Da Vitoria Lobo[2], Amrit Bedi[2], Mubarak Shah[2]

[1]University of Massachusetts Amherst
[2]University of Central Florida

Abstract

SVGs power much of the modern visual ecosystem, yet the recent wave of generative models focus almost entirely on rasterized images. We explore whether inference-time scaffolding can unlock SVG generation capabilities in off-the-shelf vision-language models (VLMs). We introduce a constrained iterative refinement framework that combines visual feedback, structured editing, and grammar-constrained decoding to address the challenges many VLMs face when generating SVGs. Across multiple VLMs and generation settings, we find that constrained decoding improves the rate of successful generation, while iterative refinement reveals a deficit in visual reasoning and self-correction. Our results highlight both the promise and current limitations of using inference-time scaffolding to adapt general-purpose VLMs for structured vector graphics generation.

CCS Concepts

• ***Computing methodologies*** → ***Computer graphics; Artificial intelligence;***

1. Introduction

Scalable Vector Graphics (SVGs) are widely used for web, interface, and illustration content because they are resolution-independent, compact, and directly editable. Despite the practical benefits of SVGs, generating them remains challenging for general-purpose vision-language models (VLMs): the model must jointly produce valid code, reason about spatial relationships, and translate precise visual details into a new medium. Existing SVG generation methods address these challenges through specialized architectures or task-specific training, decoupling them from the rapidly improving ecosystem of general-purpose multimodal models.

We investigate whether inference-time scaffolding is sufficient to make off-the-shelf VLMs effective SVG generators. In particular, we systematically evaluate iterative visual refinement, three levels of editing structure, and grammar-constrained decoding for both text-to-SVG and image-to-SVG generation across multiple model scales. Our study characterizes when these mechanisms improve syntactic reliability, how they affect visual quality, and how well current VLMs can effectively translate rendered feedback into changes to the underlying vector representation.

2. System Overview

We introduce a harness (Figure 1) that equips off-the-shelf VLMs with two complementary, inference-time mechanisms: iterative refinement based on rendered visual feedback and grammar-constrained decoding that restricts outputs to appropriate SVG and JSON syntax. Together, these mechanisms target the weaknesses of general-purpose VLMs, allowing the model to iteratively refine and correct its output while remaining bounded to a syntactically valid generation space.

Figure 1: *Our Constrained Iterative Refinement Harness.*

2.1. Iterative Refinement

Instead of generating a complete SVG in a single step, we treat generation as a sequence of iterative edits. At each step, the model observes the original user instruction, current SVG code, and its rasterization, and produces an update to the scene.

We explore three refinement variants:

Decoding	Pipeline	Text→SVG Compile↑	CLIP↑	Aesthetic↑	#Tokens	Image→SVG Compile↑	DINO↑	LPIPS↓	#Tokens
Unconstrained	Single Turn	**0.837**	**0.255**	**4.655**	1.4k	0.630	**0.704**	0.396	1.7k
	Unstructured	0.475	0.246	4.560	4.8k	0.113	**0.704**	**0.372**	10.7k
	Semi-Structured	0.180	0.205	4.415	7.9k	0.500	0.617	0.418	8.8k
	Structured	0.785	0.226	4.480	3.3k	**0.650**	0.633	0.413	8.1k
Constrained	Single Turn	**0.830**	0.220	**4.530**	0.6k	**0.830**	0.411	0.579	0.4k
	Unstructured	0.599	**0.242**	4.485	5.2k	0.647	**0.617**	0.430	5.1k
	Semi-Structured	0.512	0.206	4.445	2.7k	0.375	0.464	0.506	5.2k
	Structured	0.790	0.218	4.440	2.4k	0.695	0.603	**0.420**	3.3k

Table 1: *Results averaged across Qwen3.5 9B and 27B on MMSVG-Bench.*

Unstructured Refinement. The model generates a completely new SVG code string every iteration, fully rebuilding the image.

Semi-Structured Refinement. At each step, the model emits an action (*Add*, *Update*, *Remove*, or *Exit*) alongside a target element ID and, where relevant, an accompanying SVG code snippet.

Structured Refinement. At each step, the model emits an action (*Add*, *Update*, *Remove*, or *Exit*) alongside a target element ID and, where relevant, a fully parameterized set of element attributes.

For each variant, the model is prompted to output its reasoning along with the accompanying code/action/parameters in JSON. Unstructured iterative refinement is capped at 10 steps while the other two have up to 20 to accommodate for more granular edits.

2.2. Constrained Decoding

Many smaller models struggle to produce correct JSON structure and SVG code. We explore various inference-time constraints to address this failure mode.

For closed-source models, we constrain outputs based on JSON schemas, ensuring models adhere to the correct property names and structure. For open-source models, we further use a context-free grammar (CFG) over valid SVG structures. This restricts code generation to a subset of valid elements (e.g., `rect`, `circle`, `path`) and their respective attributes, producing syntactically valid SVGs by construction. Grammar-constrained decoding is *not* exposed by many closed-source models.

3. Results

We evaluate our framework on MMSVG-Bench [YCC*25] across multiple model scales (Qwen3.5 9B and 27B) [Qwe26], with and without grammar-constrained decoding, and across all four generation pipelines (Table 1). Metrics are computed over successfully compiled images. We provide example images generated by GPT-5.6 [Ope26] using each pipeline and decoding strategy in Figure 2.

Constrained Decoding Can Improve Compilation. We find that constrained decoding substantially improves compilation rates for most pipelines, a result consistent with our initial hypothesis.

Constrained Decoding Hurts Visual Quality. Across visual metrics and generation settings, constrained decoding consistently resulted in lower quality images. We hypothesize that enforcing our limited grammar steers token generation into unlikely and visually flawed SVG elements.

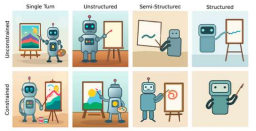


Figure 2: *Images generated by GPT-5.6 using each pipeline and decoding strategy from the Text-to-SVG prompt "A robot painting".*

Small Models Struggle With Visual Feedback. We find that for the models tested, iterative refinement has an inconsistent effect on visual quality, and often lowers compilation rates by introducing more opportunities to fail. Based on empirical results, we hypothesize these small models struggle to associate visual inconsistencies with the underlying SVG code.

4. Future Work

Several promising directions remain for improving inference-time scaffolding. Richer action spaces, alternative visual feedback mechanisms, stronger constrained decoding strategies, and prompt meta-optimization all have the possibility to improve generation quality and success rate. We also believe different evaluation metrics may better capture image quality and prompt adherence.

Beyond extending our harness, a deeper mechanistic analysis of the model's failure to incorporate visual feedback may inform improvements to general-purpose VLM spatial-reasoning capabilities.

References

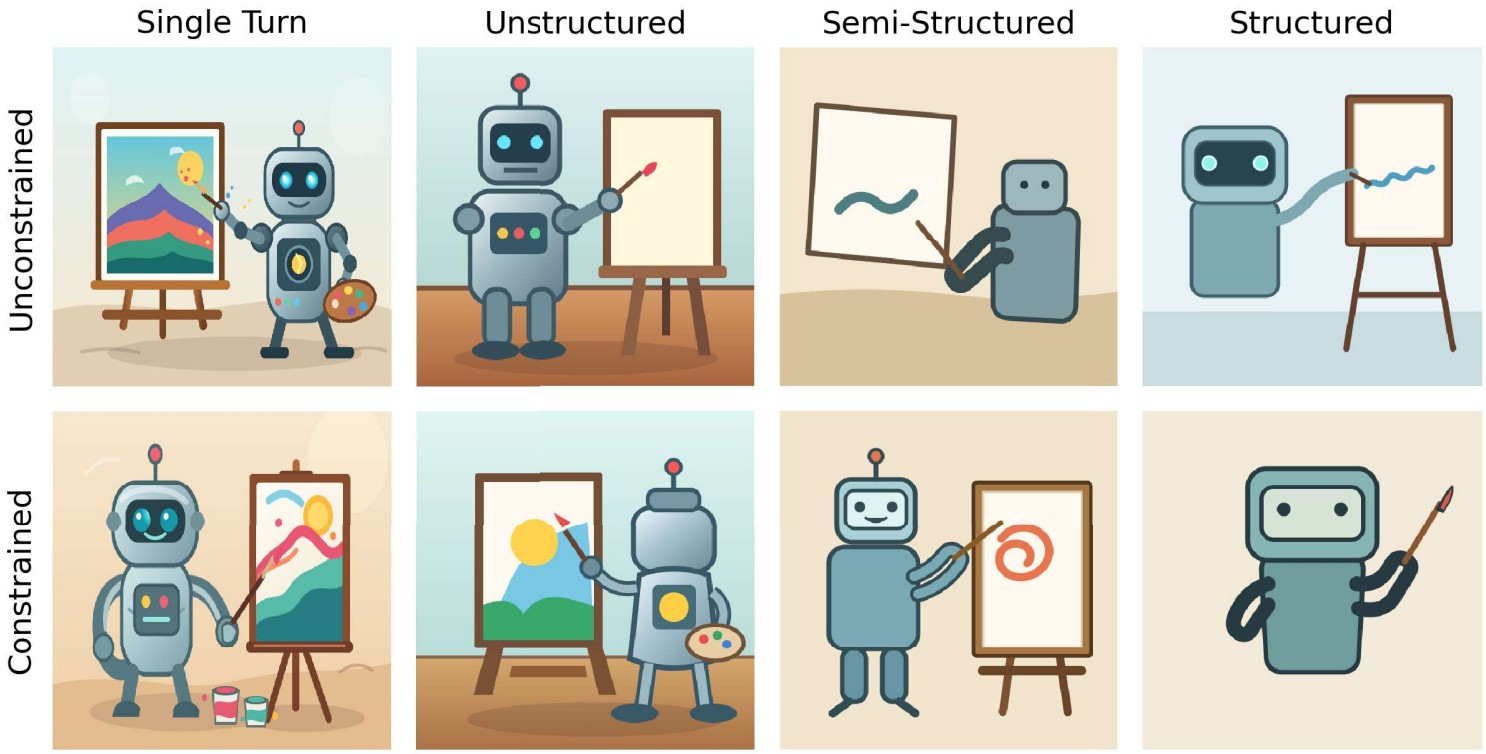
[Ope26] OpenAI: Gpt-5.6. OpenAI API, 2026 Accessed: 2026-07-13. URL: https://developers.openai.com/api/docs/models. 2

[Qwe26] Qwen Team: Qwen3.5. https://huggingface.co/Qwen, 2026. Accessed: 2026-07-13. 2

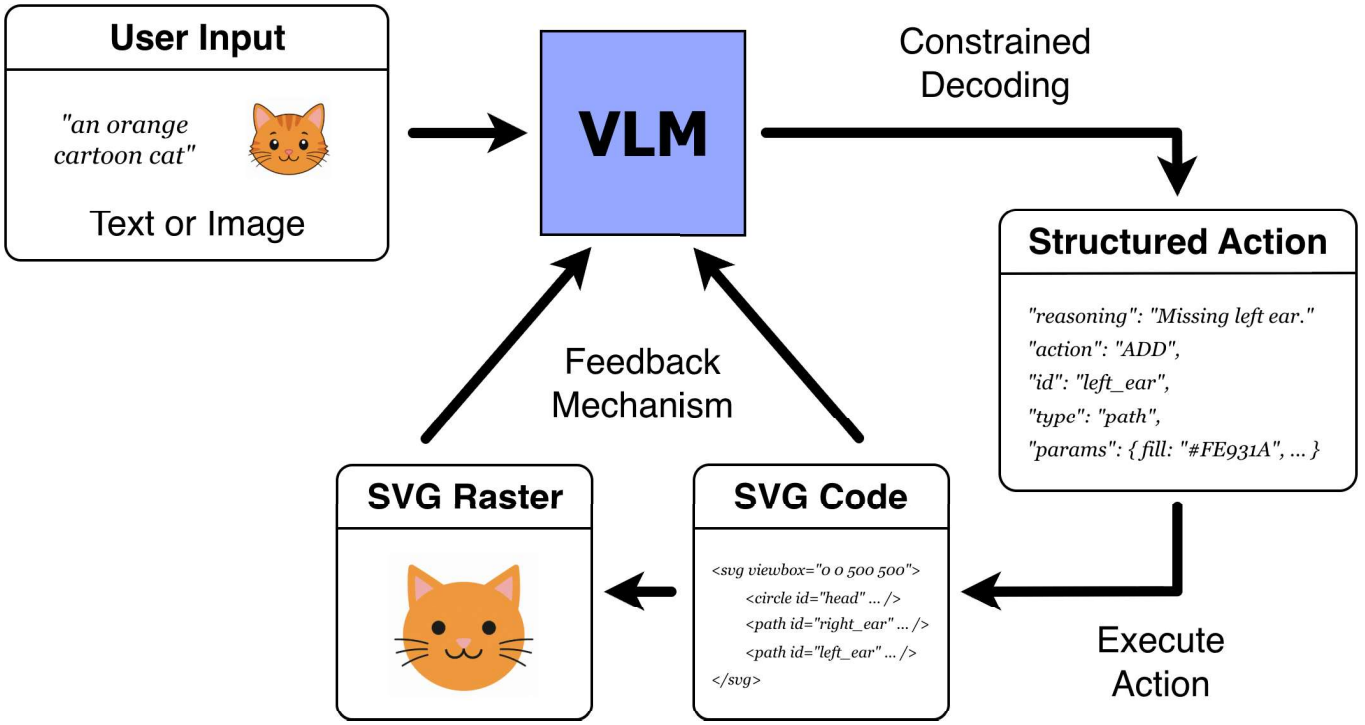
[YCC*25] Yang Y., Cheng W., Chen S., Zeng X., Yin F., Zhang J., Wang L., Yu G., Ma X., Jiang Y.-G.: Omnisvg: A unified scalable vector graphics generation model. In *Advances in Neural Information Processing Systems (NeurIPS)* (2025). 2



Qualitative Results With GPT-5.6



Finalized System Diagram



Compiled Table Results

		Text→SVG				Image→SVG			
Decoding	**Pipeline**	**Compile↑**	**CLIP↑**	**Aesthetic.↑**	**#Tokens**	**Compile↑**	**DINO↑**	**LPIPS↓**	**#Tokens**
Unconstrained	Single Turn	**0.837**	**0.255**	**4.655**	1.4k	0.630	**0.704**	0.396	1.7k
	Unstructured	0.475	0.246	4.560	4.8k	0.113	**0.704**	**0.372**	10.7k
	Semi-Structured	0.180	0.205	4.415	7.9k	0.500	0.617	0.418	8.8k
	Structured	0.785	0.226	4.480	3.3k	**0.650**	0.633	0.413	8.1k
Constrained	Single Turn	**0.830**	0.220	**4.530**	0.6k	**0.830**	0.411	0.579	0.4k
	Unstructured	0.599	**0.242**	4.485	5.2k	0.647	**0.617**	0.430	5.1k
	Semi-Structured	0.512	0.206	4.445	2.7k	0.375	0.464	0.506	3.2k
	Structured	0.790	0.218	4.440	2.4k	0.695	0.603	**0.420**	3.3k

Table 1: *Results averaged across Qwen3.5 9B and 27B on MMSVG-Bench.*

Next Steps

- Design poster
- REU writeup
- Explore additional experiments / research questions