WILDSVG: TOWARD RELIABLE SVG GENERATION UNDER REAL-WORLD CONDITIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce SVG extraction, the task of translating specific visual inputs into scalable vector graphics. Existing multimodal models such as StarVector achieve strong results when generating SVGs from clean renderings or textual descriptions, but they fall short in real-world scenarios where natural images introduce noise, clutter, and domain shifts. To address this gap, we extend StarVector's capabilities toward robust vision-to-SVG translation in the wild. A central challenge in this direction is the lack of suitable benchmarks. To fill this need, we develop two complementary datasets: Natural WildSVG, consisting of real-world images paired with SVG annotations, and Synthetic WildSVG, which integrates complex and elaborate SVG designs into real-life scenarios to simulate challenging conditions. Together, these resources provide the first foundation for systematic benchmarking SVG extraction. Building on them, we benchmark StarVector and related models. Our study establishes SVG extraction as a new problem domain, introduces datasets and evaluation protocols for its study, taking initial steps toward extending multimodal LLMs to handle reliable SVG generation in complex, natural scenes.

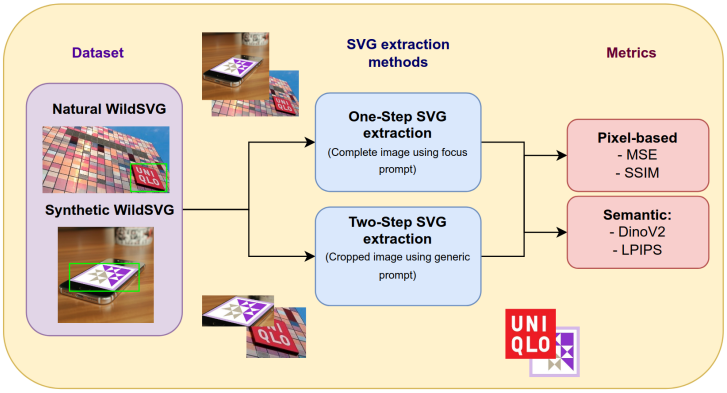

Figure 1: **Overview of WildSVG benchmark**. We can observe both WildSVG datasets and SVG extraction approaches. This benchmark is novel, introducing datasets specifically designed for the *SVG extraction* problem—a task not previously addressed in existing research. As well as addressing limitations of unique metrics evaluation protocols, by combining both semantic and fidelity-based metrics for comprehensive assessment.

1 INTRODUCTION

Scalable Vector Graphics (SVGs) are an XML-based open standard and the leading format for vector graphic representation Quint (2003), widely adopted in modern image rendering. However, efficiently generating SVGs remains a significant challenge, as the format supports a wide range of primitives, from basic curves such as *path* to more complex shapes like *ellipse* or *polygon*. The task of image vectorization—producing SVG code from rendered images—remains unsolved by

the industry. Traditional approaches rely on complex path operations, while deep learning methods struggle to generalize and often underutilize higher-level SVG primitives.

Recent advances, such as StarVector Rodriguez et al. (2025a), have demonstrated the potential of multimodal large language models (MLLMs) for SVG generation. StarVector, trained on SVG-Stack—a dataset of over two million samples—achieved state-of-the-art results through a reinforcement learning pipeline with visual feedback Rodriguez et al. (2025b). Despite this progress, existing methods are limited to controlled inputs such as clean renderings or text prompts. They fail when confronted with the challenges of natural images, where SVG elements are embedded in cluttered, noisy, and context-rich environments.

To address this gap, we introduce the SVG extraction task, which focuses on identifying graphical elements such as logos, icons, or pictograms within real-world images—given user guidance—and generating their corresponding SVG code. Unlike full-image vectorization, SVG extraction requires selective abstraction: isolating target elements while ignoring irrelevant visual content such as textures, shadows, occlusions, and perspective distortions.

We take three key steps to establish SVG extraction as a research problem. First, we introduce *WildSVG*, the first benchmark for this task, composed of two complementary datasets: (i) *Natural WildSVG*, which grounds vector annotations in real-world images, and (ii) *Synthetic WildSVG*, which embeds complex SVGs into natural scenes to simulate challenging visual conditions. Second, we define evaluation protocols to support consistent and fair benchmarking across models. Finally, we adopt StarVector and other multimodal models as baselines, establishing initial performance levels and highlighting open challenges. Together, these steps lay the foundation for systematic study of SVG extraction.

2 Related work

Research on SVG generation is still in its early stages, with most work focused on reconstructing full graphics from clean renderings Rodriguez et al. (2025a). To the best of our knowledge, no prior work has addressed the *SVG extraction* task—isolating graphical elements from natural images and generating structured vector representations. Nevertheless, two research directions are closely related and inform our setting: (1) logo detection, and (2) image-to-SVG generation.

2.1 Logo detection

Logo detection, a specialized form of object detection, has been widely studied due to applications in multimedia analysis, brand monitoring, and copyright protection. Early approaches relied on hand-crafted features combined with classifiers, while the rise of deep learning established detectors such as YOLO Khanam & Hussain (2024), DETR Carion et al. (2020), and the R-CNN family He et al. (2018) as the standard. Despite their success, these methods face challenges with dataset imbalance and the closed-set assumption, which limit their ability to generalize to unseen logos Hou et al. (2023).

Recent work explores zero-shot and open-vocabulary detection to address these issues by leveraging language–vision alignment. For example, some methods replace fixed labels with textual descriptions Zareian et al. (2021), while others combine CLIP-based classifiers with object-agnostic detectors Shulgin & Makarov (2023) or employ transformer-based region embeddings Minderer et al. (2022); Gu et al. (2022). Multimodal LLMs Chen et al. (2023); Bai et al. (2025); Deitke et al. (2024) further integrate such tasks into pretraining, extending them toward more context-aware object detection Zang et al. (2025); Yin et al. (2025). However, these approaches output bounding boxes or class labels only, whereas SVG extraction requires both localization and structured vector code generation.

2.2 SVG generation

Traditional vectorization methods rely on geometric fitting with the *path* primitive Wu et al. (2023a); Weber (2025); Pun & Tang (2025), often producing verbose SVG code with limited structural abstraction. Latent-variable models Jain et al. (2022); Carlier et al. (2020); Ma et al. (2022); lat

(2024) increase flexibility but are typically constrained to narrow SVG subsets and yield non-human-readable outputs.

Recent advances expand into specialized domains such as emoji generation Wang & Lian (2021); Lopes et al. (2019); Wu et al. (2023b), or employ LLMs for SVG creation and editing Bubeck et al. (2023); Cai et al. (2024), framing vector graphics as structured program synthesis Chen et al. (2021); Feng et al. (2020). The most significant development is StarVector Rodriguez et al. (2025a), which casts SVG generation as multimodal inverse rendering and code generation, trained on the large-scale SVG-Stack dataset. A posterior reinforcement learning extension, with rendering feedback (RLRF), further improved its visual fidelity Rodriguez et al. (2025b). Yet, these models remain restricted to clean renderings and degrade substantially in natural images with clutter, occlusion, or noise.

2.3 Dataset survey

The task of identifying and processing SVGs within real-world images requires new datasets, as none currently address this problem. The original StarVector paper Rodriguez et al. (2025a) introduced SVG-Stack, along with several subsets, as resources for the Image-to-SVG task. While valuable, SVG-Stack focuses on clean renderings and does not capture the challenges of detecting and generating SVGs from natural contexts. Conversely, logo detection datasets provide only localization information (e.g., bounding boxes) without vector annotations.

Table 1: Overview of logo detection datasets

Name	Classes	Images	Objects	Use case	Dataset origin	License
Belgalogos Joly & Buisson (2009)	37	10,000	2,695	Manually selected and annotated images from photo-journalist archives	General logos in a wide range of events present in the press	Copyrighted, academic use only
FlickrLogos-27 Kalantidis et al. (2011)	27	1,080	4,671	General logo dataset focused on real-life scenarios	Manually created from Flickr image search, logo selection and annotation	Copyrighted, follow Flickr terms
FlickrLogos-32 Romberg et al. (2011)	32	2,240	5,644	General logo dataset	Manually created from Flickr image search, logo selection and annotation	Copyrighted, follow Flickr terms
SportLogo Kuznetsov & Savchenko (2020)	31	2,836	-	Sports logos, primarily NHL and NBA	Manually collected via search engine	Creative Commons Attribution 4.0
Logos-in-the-Wild Tüzkö et al. (2017)	871	11,054	32,850	Logos in real-world scenarios	Manually selected and annotated Google image search results	Copyrighted, fair use
QMUL-OpenLogo Su et al. (2018)	352	27,083	-	Merged from 7 logo detection datasets	Diverse logo conditions, curated for variation in scale and context	Research use only
FoodLogoDet-1500 Hou et al. (2021)	1,500	99,768	145,400	Logos in the food industry	Curated list, auto-searched then manually annotated	Not disclosed
LogoDet-3K Wang et al. (2022)	3,000	158,652	194,261	General-purpose logos	Manually constructed from web-crawled images	Not disclosed

We reviewed existing publicly available logo detection datasets (Table 1). Most provide bounding boxes in natural or semi-natural settings but lack vectorized logo representations. Among them, Logos-in-the-Wild stands out for its scale and diversity, covering difficult real-world conditions such as perspective distortion, scale variation, occlusion, and noisy textures. For SVG generation datasets, we focus on SVG-Stack and its subsets (Table 2), which remain the most comprehensive and high-quality resources for vector graphics research. However, they do not include SVGs embedded in real-world image contexts.

Taken together, no existing dataset satisfies the requirements of SVG extraction: grounding vector graphics within natural scenes while maintaining structured SVG annotations. This gap directly motivates the creation of our WildSVG dataset, introduced in the following section.

Table 2: Overview of SVG generation datasets

Name	Train	Validation	Test	Primitives	Annotations	Images
SVG Stack	2,1M	108k	5,7k	All	Caption	SVG render
SVG Diagrams	-	-	472	All	Caption	SVG render
SVG Fonts	1,8M	91,5k	4,8k	Vector Path	Font Type	SVG render
SVG Emoji	8,7k	667	668	All	Class	SVG render
SVG Icons	80,4k	6,2k	2,4k	Vector Path	Class, Caption	SVG render

2.4 Motivation for SVG extraction

Prior work in logo detection and SVG generation leaves a clear gap. Detection models can localize target regions but cannot produce structured vector outputs, while SVG generation models excel on synthetic renderings but fail in real-world conditions. The *SVG extraction task* bridges these domains, requiring both localization and vector generation. To enable systematic study of this task, we introduce the WildSVG benchmark, which provides the first datasets designed specifically for SVG extraction in natural scenes.

3 WildSVG datasets

To enable systematic study of the SVG extraction task, we introduce the WildSVG datasets, consisting of two complementary datasets: *Natural WildSVG* and *Synthetic WildSVG*. Together, they combine the realism of naturally occurring logos with the diversity and controllability of synthetic SVG integration. The dataset generation pipelines are illustrated in Appendix Figures 6 and 7.

3.1 Natural WildSVG

Built from Logos-in-the-Wild Tüzkö et al. (2017), Natural WildSVG augments logo detections with vectorized annotations. Each bounding box is paired with (i) an SVG retrieved from worldvectorlogo.com, (ii) a textual description, and (iii) a focus prompt specifying the target element. To ensure consistency, candidate SVGs were validated using a VLLM-based judging model and ranked by DINOv2 features similarity,Oquab et al. (2023), between rasterized SVGs and cropped detections. This process produced high-quality matches between natural logo appearances and their corresponding vector representations.

3.2 Synthetic WildSVG

To complement natural logos, Synthetic WildSVG integrates complex SVGs into realistic scenes. Starting from SVG-Stack Rodriguez et al. (2025a), each SVG and its textual description were used to generate synthetic compositions with `gemini-2.0-flash-preview-image-generation` (now known as *Nano Banana*). Prompts were manually optimized to preserve SVG fidelity while embedding the logos naturally in the background (Appendix Fig. 8). We additionally generated focus prompts for the complete dataset and manually annotated bounding boxes for the test split to support reliable evaluation. This dataset introduces diverse and complex SVG types under controlled but visually challenging conditions.

3.3 QUALITY FILTERING AND RESULTING DATASET

Both datasets underwent automated filtering inspired by StarVector-RL scoring system Rodriguez et al. (2025b). Each sample was scored on *constancy* (SVG–image similarity), *alignment* (focus prompt accuracy), and, for synthetic data, *aesthetics* (realism of integration). We combined these into aggregate scores, prioritizing constancy and aesthetics over alignment (details in Appendix Figures 9 and 12, and Equation 1). The resulting dataset statistics are shown in Table 3.

The final datasets are smaller than SVG-Stack or Logos-in-the-Wild, reflecting their intended use as fine-tuning and evaluation benchmarks rather than pretraining corpora. Despite API constraints during synthetic generation, the resulting resources balance natural complexity and synthetic diversity, establishing WildSVG as the first benchmark for SVG extraction.

Table 3: WildSVG datasets

Dataset	Train	Validation	Test	Primitives	Annotations
Natural WildSVG	12759	1418	227	Path	Logo brand, focus prompt, description, bounding box
Synthetic WildSVG	2104	190	99	All	Focus prompt, description

3.4 LICENSING

The SVG-Stack data is released under a Creative Commons Attribution 4.0 International (CC BY 4.0) license. Accordingly, our Synthetic WildSVG extension is distributed under the same license, permitting both research and commercial use. In contrast, due to the copyrighted nature of images employed by Logos-in-the-Wild, the Natural WildSVG extension can only be licensed for research purposes.

4 WILDSVG BENCHMARK

The WildSVG benchmark aims to evaluate model performance on the SVG extraction task in both *natural* scenarios, which involve complex detections and real-world noise, and *synthetic* scenarios, which feature simpler visual noise but more complex SVG structures. For this purpose, we employ the test split of both datasets.

To establish a fair baseline for SVG extraction, we report four complementary metrics:

- **L2 distance** and **SSIM** – pixel-level fidelity
- **LPIPS** and **DINO score** – perceptual and semantic similarity

The use of multiple metrics is motivated by their complementary strengths. Pixel-level fidelity metrics (L2, SSIM) ensure precise reproduction of visual details, but they may penalize outputs that are perceptually faithful yet not perfectly aligned at the pixel level. Conversely, perceptual and semantic metrics (LPIPS, DINOv2) Oquab et al. (2023); Zhang et al. (2018) capture higher-level similarity, complementing fidelity-based measures. Together, these metrics provide a more comprehensive evaluation, balancing strict accuracy with semantic consistency.

4.1 EVALUATED MODELS

As a contribution to the SVG extraction task, we establish baseline results across a range of recent VLLM families, including StarVector. The following models were evaluated:

1. **Qwen**: Qwen2.5VL-72B-Instruct
2. **Gemini**: Gemini 2.0 Flash, Gemini Flash 2.5
3. **Claude**: Claude Opus 4, Claude Opus 4.1

4. **GLM**: GLM-4.1V-9B-Thinking, z-ai GLM-4.5V
5. **GPT**: GPT-4.1, GPT-5
6. **StarVector**: rlvg-7b-long-context

Our goal is to analyze both the performance and trade-offs of each approach. Some models, such as GPT variants, employ dense Transformer architectures, while others incorporate mixture-of-experts (MoE) designs. We additionally investigate the impact of different visual encoders on extraction performance, as well as differences between open-source and closed-source models.

4.2 Evaluation settings

For a comprehensive evaluation, we conduct SVG extraction under two setups:

1. **Full-image extraction with focus prompt:** The model receives the complete image along with a focus prompt that specifies the target element for extraction.
2. **Two-step extraction with perfect object detection:** Images are first cropped using ground-truth bounding boxes before SVG generation. This setting reduces distractors, helping models—particularly StarVector—focus on specific features. It also serves as an upper bound for two-step methods that rely on external detection modules.

5 Benchmark results

From our current benchmark, we present reduced tables, Table 4 and 5, containing the most recent model of each family; the complete results are provided in Appendix Tables 6 and 7. Since models within a family generally exhibit similar behavior, we focus on the most recent and best-performing representatives.

Two clear trends emerge from the results. First, across families, models produce SVGs that follow broadly similar patterns, with only a few notable outliers. For example, StarVector frequently attempts to render the entire image rather than isolating the SVG. Other specific cases include Claude Opus 4 misrepresenting the FedEx logo (Figure 4) and the GLM family struggling with the human-shaped SVG (Figure 3). Second, models consistently achieve higher scores on the synthetic dataset, reflecting the greater difficulty of the natural dataset, where scaling, perspective distortion, occlusions, shadows, and noise complicate vectorization. As shown in Figure 2, even advanced models struggle with ambiguous cases such as the Special K box, where only GPT-5 partially captures the "K" logo.

Table 4: VLLM benchmark for one-step SVG extraction task

	Natural	**Synthetic**
Model	**L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓**	**L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓**
Qwen2.5VL-72B-Instruct	0.22 / 0.58 / 0.77 / 0.41	0.21 / 0.58 / 0.77 / 0.42
Gemini Flash 2.5	0.20 / 0.58 / 0.79 / 0.42	0.21 / 0.57 / 0.78 / 0.43
Starvector rlvg-7b-long-context	0.15 / 0.63 / 0.69 / 0.39	0.16 / 0.61 / 0.76 / 0.43
Claude Opus 4.1	**0.19 / 0.61 / 0.80 / 0.40**	**0.20 / 0.58 / 0.80 / 0.42**
z-ai GLM 4.5V	0.18 / 0.61 / 0.79 / 0.39	0.19 / 0.59 / 0.77 / 0.40
GPT 5	*0.19 / 0.58 / 0.80 / 0.40*	*0.22 / 0.57 / 0.79 / 0.42*

Detection itself does not consistently improve results across families. The notable exception is StarVector, which in the one-step setting ignores the prompt and attempts to vectorize the full image

[2]POD standing for Perfect Object Detection

Table 5: VLLM benchmark for two-step perfect logo detection SVG extraction task

Model	Natural L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓	Synthetic L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓
Qwen2.5VL-72B-Instruct	0.21 / 0.62 / 0.81 / 0.36	0.20 / 0.61 / 0.85 / 0.34
Gemini Flash 2.5	0.19 / 0.64 / 0.85 / 0.32	0.19 / 0.64 / 0.88 / 0.33
Starvector rlvg-7b-long-context	0.18 / 0.60 / 0.74 / 0.46	0.16 / 0.63 / 0.82 / 0.37
Claude Opus 4.1	*0.16 / 0.66 / 0.86 / 0.32*	**0.16 / 0.65 / 0.90 / 0.30**
z-ai GLM 4.5V	0.20 / 0.63 / 0.83 / 0.34	0.18 / 0.64 / 0.86 / 0.32
GPT 5	**0.18 / 0.63 / 0.87 / 0.34**	*0.18 / 0.63 / 0.89 / 0.31*

(Figure 2). Despite the fact than other VLLM families, including Qwen, do have the capacity to focus on the given prompt. This behavior suggests a weakness in StarVector's training pipeline, as text-to-SVG and image-to-SVG tasks are learned separately in this regime which may reduce the alignment between prompt and image features during SVG generation.

Figure 2: Comparison of VLLMs for one-step SVG extraction natural dataset

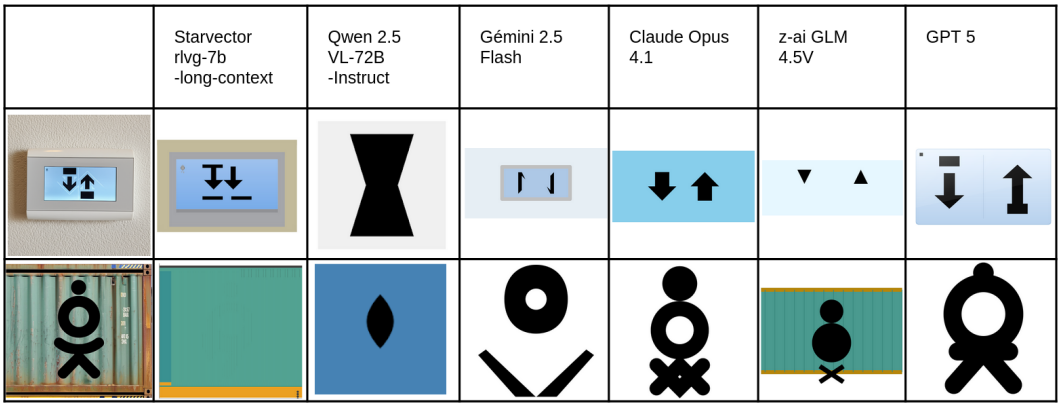

Figure 3: Comparison of VLLMs for one-step SVG extraction synthetic dataset

Across families, VLLMs generally optimize for semantic similarity rather than aesthetic fidelity. This tendency is reflected in the frequent use of the *text* primitive to approximate letters with similar fonts, rather than rendering them as shapes. As a result, SVGs achieve strong performance on semantic metrics (LPIPS, DINO) but weaker performance on pixel-level metrics (L2, SSIM). For instance, GPT-5's rendering of the Heineken logo (Figure 4) appears convincing in overall structure but reveals clear inaccuracies upon closer examination. Synthetic examples further highlight this trade-off: some models reproduce SVGs with structures reminiscent of the original, but insufficiently faithful for precise extraction (Figures 3, 5). Among all families, Claude 4.1 and GPT-5 deliver the most semantically consistent and highest-fidelity SVGs, though both remain below the fidelity required for a complete solution of SVG extraction.

StarVector diverges from this general trend, sacrificing semantic fidelity in favor of visual aesthetics. This is reflected in the metrics, where L2 and SSIM are prioritized over LPIPS and DINO, and in qualitative examples such as the FedEx and Heineken logos (Figure 4). In these cases, StarVector often relies on shape primitives to render letters, a strategy preferable to text primitives since reproducing exact font styles, kerning, and spacing is effectively impossible. While this approach reduces semantic scores, it has the potential to produce SVGs visually closer to the original designs. Nonetheless, the outputs remain inconsistent: for example, the synthetic POD logo (Figure 5) demonstrates significant misalignment in element positioning and structure. For the specific task of SVG extraction, StarVector's results remain below the semantic and aesthetic quality achieved by Claude and GPT models. However, StarVector shows strong capabilities in generating complex SVGs directly from rasterized SVGs. We hypothesize that noise in real-world scenarios—such as textures and shadows—may overwhelm the model, leading it to overfit to subtle visual variations rather than isolating the core logo structure.

Overall, our baseline demonstrates that current VLLMs can generate SVGs that are semantically meaningful but still fall short in aesthetic fidelity. Across families, most models achieve relatively similar scores regardless of the visual encoder or LLM architecture, suggesting that model size plays a greater role than design choices. As Open-source models, typically ranging from 10–70B parameters, tend to perform slightly worse than larger proprietary systems. However, even the strongest models, such as Claude 4.1 and GPT-5, plateau at approximately DINO 90, LPIPS 30, SSIM 60, and L2 15. By comparison, achieving high-fidelity SVG generation would require scores closer to DINO 95, LPIPS 10, SSIM 80, and L2 9. These results point to a performance ceiling in current approaches.

Key findings:

- Synthetic dataset is consistently easier than Natural.
- Models optimize for semantic similarity (LPIPS, DINO) over pixel fidelity (L2, SSIM).
- StarVector diverges from other families, favoring aesthetics over semantics.
- Even strongest models plateau below high-fidelity thresholds, leaving clear headroom for future work.

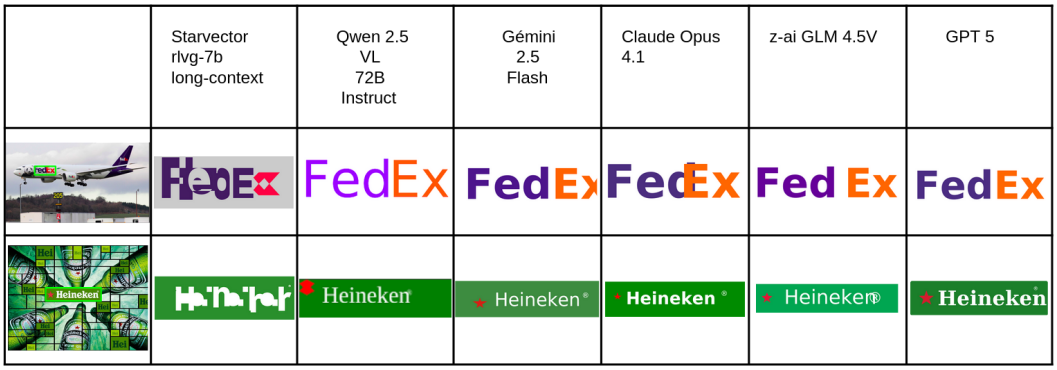

Figure 4: Comparison of VLLMs for two-step SVG extraction natural dataset

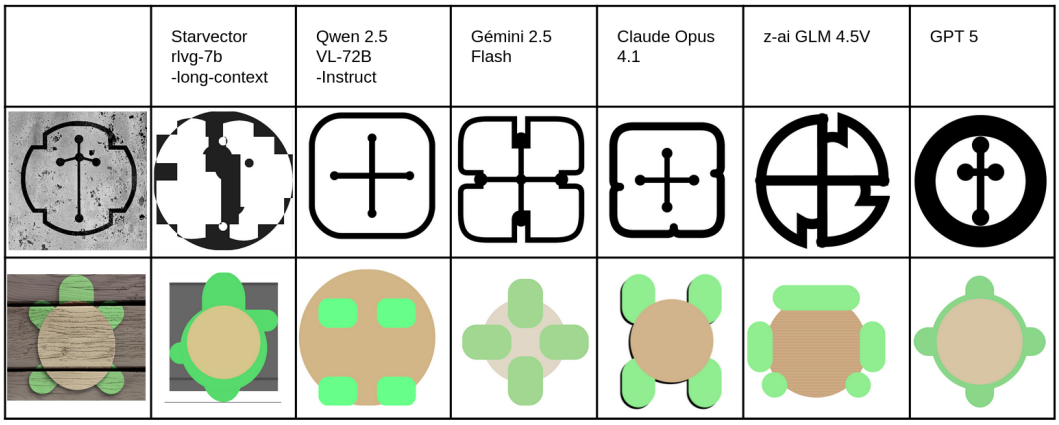

Figure 5: Comparison of VLLMs for two-step SVG extraction synthetic dataset

6 Conclusion

We introduced the task of **SVG extraction**, extending multimodal models to generate vector graphics directly from natural images, and proposed **WildSVG**, the first benchmark for this problem. WildSVG combines real-world logos with synthetic compositions, enabling evaluation under both natural and controlled conditions.

Our benchmarking of leading VLLM families reveals three consistent takeaways: (1) models perform better on synthetic than natural data, showing the impact of real-world distortions; (2) current systems prioritize semantic similarity over pixel fidelity; (3) even the strongest models plateau below high-fidelity thresholds, leaving clear headroom for improvement. While Claude and GPT balance fidelity and semantics most effectively, StarVector highlights a contrasting trade-off by favoring aesthetics over semantics.

Looking ahead, we identify several open research directions: (i) improving alignment between prompts and structured vector outputs, particularly for StarVector; (ii) integrating SVG generation and extraction tasks into VLLM training pipelines to improve fidelity in vector code; (iii) extending two-step approaches to leverage SVG generation without requiring task-specific fine-tuning. and (iiii) expanding WildSVG datasets to allow broader training approaches not only finetuning.

By framing SVG extraction as a benchmarked task, we aim to catalyze future research at the intersection of vision, language, and structured graphics generation.

References

Representation learning for continuous vector graphics, September 2024. URL https://www.freepatentsonline.com/y2024/0303870.html.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. URL https://arxiv.org/abs/2303.12712.

Mu Cai, Zeyi Huang, Yuheng Li, Utkarsh Ojha, Haohan Wang, and Yong Jae Lee. Leveraging large language models for scalable vector graphics-driven image understanding, 2024. URL https://arxiv.org/abs/2306.06094.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. URL https://arxiv.org/abs/2005.12872.

Alexandre Carlier, Martin Danelljan, Alexandre Alahi, and Radu Timofte. Deepsvg: A hierarchical generative network for vector graphics animation, 2020. URL https://arxiv.org/abs/2007.11301.

Mark Chen, Jerry Tworek, Heewoo Jun, and etc.. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, Siamak Shakeri, Mostafa Dehghani, Daniel Salz, Mario Lucic, Michael Tschannen, Arsha Nagrani, Hexiang Hu, Mandar Joshi, Bo Pang, Ceslee Montgomery, Paulina Pietrzyk, Marvin Ritter, AJ Piergiovanni, Matthias Minderer, Filip Pavetic, Austin Waters, Gang Li, Ibrahim Alabdulmohsin, Lucas Beyer, Julien Amelot, Kenton Lee, Andreas Peter Steiner, Yang Li, Daniel Keysers, Anurag Arnab, Yuanzhong Xu, Keran Rong, Alexander Kolesnikov, Mojtaba Seyedhosseini, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. Pali-x: On scaling up a multilingual vision and language model, 2023. URL https://arxiv.org/abs/2305.18565.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models, 2024. URL https://arxiv.org/abs/2409.17146.

Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. Codebert: A pre-trained model for programming and natural languages, 2020. URL https://arxiv.org/abs/2002.08155.

Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation, 2022. URL https://arxiv.org/abs/2104.13921.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2018. URL https://arxiv.org/abs/1703.06870.

Qiang Hou, Weiqing Min, Jing Wang, Sujuan Hou, Yuanjie Zheng, and Shuqiang Jiang. Foodlogodet-1500: A dataset for large-scale food logo detection via multi-scale feature decoupling network. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, pp. 4670–4679, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450386517. doi: 10.1145/3474085.3475289. URL https://doi.org/10.1145/3474085.3475289.

Sujuan Hou, Jiacheng Li, Weiqing Min, Qiang Hou, Yanna Zhao, Yuanjie Zheng, and Shuqiang Jiang. Deep learning for logo detection: A survey. *ACM Trans. Multimedia Comput. Commun. Appl.*, 20(3), October 2023. ISSN 1551-6857. doi: 10.1145/3611309. URL https://doi.org/10.1145/3611309.

Ajay Jain, Amber Xie, and Pieter Abbeel. Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models, 2022. URL https://arxiv.org/abs/2211.11319.

Alexis Joly and Olivier Buisson. Logo retrieval with a contrario visual query expansion. In *Proceedings of the 17th ACM International Conference on Multimedia*, MM '09, pp. 581–584, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605586083. doi: 10.1145/1631272.1631361. URL https://doi.org/10.1145/1631272.1631361.

Yannis Kalantidis, Lluis Garcia Pueyo, Michele Trevisiol, Roelof van Zwol, and Yannis Avrithis. Scalable triangulation-based logo recognition. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ICMR '11, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450303361. doi: 10.1145/1991996.1992016. URL https://doi.org/10.1145/1991996.1992016.

Rahima Khanam and Muhammad Hussain. Yolov11: An overview of the key architectural enhancements, 2024. URL https://arxiv.org/abs/2410.17725.

Andrey Kuznetsov and Andrey V. Savchenko. A new sport teams logo dataset for detection tasks. In Leszek J. Chmielewski, Ryszard Kozera, and Arkadiusz Orłowski (eds.), *Computer Vision and Graphics*, pp. 87–97, Cham, 2020. Springer International Publishing. ISBN 978-3-030-59006-2.

Raphael Gontijo Lopes, David Ha, Douglas Eck, and Jonathon Shlens. A learned representation for scalable vector graphics, 2019. URL https://arxiv.org/abs/1904.02632.

Xu Ma, Yuqian Zhou, Xingqian Xu, Bin Sun, Valerii Filev, Nikita Orlov, Yun Fu, and Humphrey Shi. Towards layer-wise image vectorization, 2022. URL https://arxiv.org/abs/2206.04655.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 728–755, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-20080-9.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Sanford Pun and Chris Tang. Vision cortex. vtracer, 2025. URL https://www.visioncortex.org/vtracer-docs.

Antoine Quint. Scalable vector graphics. *IEEE MultiMedia*, 10(3):99–102, July 2003. ISSN 1070-986X. doi: 10.1109/MMUL.2003.1218261. URL https://doi.org/10.1109/MMUL.2003.1218261.

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, 2025a. URL https://arxiv.org/abs/2312.11556.

Juan A. Rodriguez, Haotian Zhang, Abhay Puri, Aarash Feizi, Rishav Pramanik, Pascal Wichmann, Arnab Mondal, Mohammad Reza Samsami, Rabiul Awal, Perouz Taslakian, Spandana Gella, Sai Rajeswar, David Vazquez, Christopher Pal, and Marco Pedersoli. Rendering-aware reinforcement learning for vector graphics generation, 2025b. URL https://arxiv.org/abs/2505.20793.

Stefan Romberg, Lluis Garcia Pueyo, Rainer Lienhart, and Roelof van Zwol. Scalable logo recognition in real-world images. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ICMR '11, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450303361. doi: 10.1145/1991996.1992021. URL https://doi.org/10.1145/1991996.1992021.

Mikhail Shulgin and Ilya Makarov. Scalable zero-shot logo recognition. *IEEE Access*, 11:142702–142710, 2023. doi: 10.1109/ACCESS.2023.3342721.

Hang Su, Xiatian Zhu, and Shaogang Gong. Open logo detection challenge, 2018. URL https://arxiv.org/abs/1807.01964.

Andras Tüzkö, Christian Herrmann, Daniel Manger, and Jürgen Beyerer. Open set logo detection and retrieval, 2017. URL https://arxiv.org/abs/1710.10891.

Jing Wang, Weiqing Min, Sujuan Hou, Shengnan Ma, Yuanjie Zheng, and Shuqiang Jiang. Logodet-3k: A large-scale image dataset for logo detection. *ACM Trans. Multimedia Comput. Commun. Appl.*, 18(1), January 2022. ISSN 1551-6857. doi: 10.1145/3466780. URL https://doi.org/10.1145/3466780.

Yizhi Wang and Zhouhui Lian. Deepvecfont: Synthesizing high-quality vector fonts via dual-modality learning, 2021. URL https://arxiv.org/abs/2110.06688.

Martin Weber. Autotrace, 2025. URL https://github.com/autotrace/autotrace.

Ronghuan Wu, Wanchao Su, Kede Ma, and Jing Liao. Iconshop: Text-guided vector icon synthesis with autoregressive transformers, 2023a. URL https://arxiv.org/abs/2304.14400.

Ronghuan Wu, Wanchao Su, Kede Ma, and Jing Liao. Iconshop: Text-guided vector icon synthesis with autoregressive transformers, 2023b. URL https://arxiv.org/abs/2304.14400.

Heng Yin, Yuqiang Ren, Ke Yan, Shouhong Ding, and Yongtao Hao. Rod-mllm: Towards more reliable object detection in multimodal large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 14358–14368, June 2025.

Yuhang Zang, Wei Li, Jun Han, Kaiyang Zhou, and Chen Change Loy. Contextual object detection with multimodal large language models. *International Journal of Computer Vision*, 133(2):825–843, Feb 2025. ISSN 1573-1405. doi: 10.1007/s11263-024-02214-4. URL https://doi.org/10.1007/s11263-024-02214-4.

Alireza Zareian, Kevin Dela Rosa, Derek Hao Hu, and Shih-Fu Chang. Open-vocabulary object detection using captions, 2021. URL https://arxiv.org/abs/2011.10678.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

A Appendix

A.1 LLM Usage

Parts of this manuscript were refined and polished using ChatGPT (GPT-5), a large language model developed by OpenAI. The model was employed solely for language editing and clarity improvements; all technical content, data analyses, and conceptual contributions remain the original work of the authors.

A.2 Complete benchmark

In Tables 6 and 7, we report the benchmark results for all VLLMs evaluated in our study.

Table 6: VLLM benchmark for one-step SVG extraction task

Model	Natural L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓	Synthetic L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓
Qwen2.5VL-72B-Instruct	0.22 / 0.58 / 0.77 / 0.41	0.21 / 0.58 / 0.77 / 0.42
Gemini Flash 2	0.17 / 0.61 / 0.78 / 0.38	0.19 / 0.63 / 0.83 / 0.32
Gemini Flash 2.5	0.20 / 0.58 / 0.79 / 0.42	0.21 / 0.57 / 0.78 / 0.43
Starvector rlvg-7b-long-context	0.15 / 0.63 / 0.69 / 0.39	0.16 / 0.61 / 0.76 / 0.43
Claude Opus 4	0.18 / 0.60 / 0.78 / 0.40	0.19 / 0.62 / 0.84 / 0.34
Claude Opus 4.1	**0.19 / 0.61 / 0.80 / 0.40**	**0.20 / 0.58 / 0.80 / 0.42**
GLM-4.1V-9B-Thinking	0.17 / 0.63 / 0.75 / 0.37	0.21 / 0.62 / 0.78 / 0.37
z-ai GLM 4.5V	0.18 / 0.61 / 0.79 / 0.39	0.19 / 0.59 / 0.77 / 0.40
GPT 4.1	0.18 / 0.59 / 0.81 / 0.39	0.18 / 0.64 / 0.86 / 0.32
GPT 5	*0.19 / 0.58 / 0.80 / 0.40*	*0.22 / 0.57 / 0.79 / 0.42*

Table 7: VLLM benchmark for two-step, perfect logo detection, SVG extraction task

Model	Natural L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓	Synthetic L2 ↓ / SSIM ↑ / DINO ↑ / LPIPS ↓
Qwen2.5VL-72B-Instruct	0.21 / 0.62 / 0.81 / 0.36	0.20 / 0.61 / 0.85 / 0.34
Gemini Flash 2	0.18 / 0.61 / 0.76 / 0.40	0.18 / 0.64 / 0.86 / 0.32
Gemini Flash 2.5	0.19 / 0.64 / 0.85 / 0.32	0.19 / 0.64 / 0.88 / 0.33
Starvector rlvg-7b-long-context	0.18 / 0.60 / 0.74 / 0.46	0.16 / 0.63 / 0.82 / 0.37
Claude Opus 4	0.19 / 0.58 / 0.78 / 0.43	0.15 / 0.66 / 0.88 / 0.30
Claude Opus 4.1	*0.16 / 0.66 / 0.86 / 0.32*	**0.16 / 0.65 / 0.90 / 0.30**
GLM-4.1V-9B-Thinking	0.20 / 0.59 / 0.76 / 0.41	0.18 / 0.63 / 0.83 / 0.33
z-ai GLM 4.5V	0.20 / 0.63 / 0.83 / 0.34	0.18 / 0.64 / 0.86 / 0.32
GPT 4.1	0.20 / 0.57 / 0.80 / 0.41	0.17 / 0.63 / 0.88 / 0.31
GPT 5	**0.18 / 0.63 / 0.87 / 0.34**	*0.18 / 0.63 / 0.89 / 0.31*

A.3 Dataset generation

A.3.1 Dataset generation pipelines

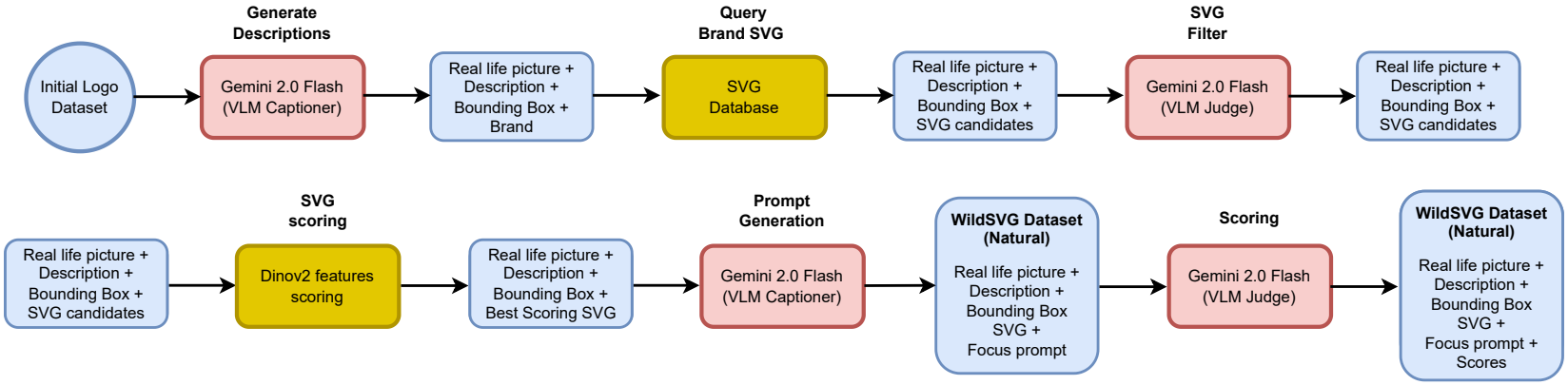

Figure 6: Pipeline for synthetic WildSVG generation

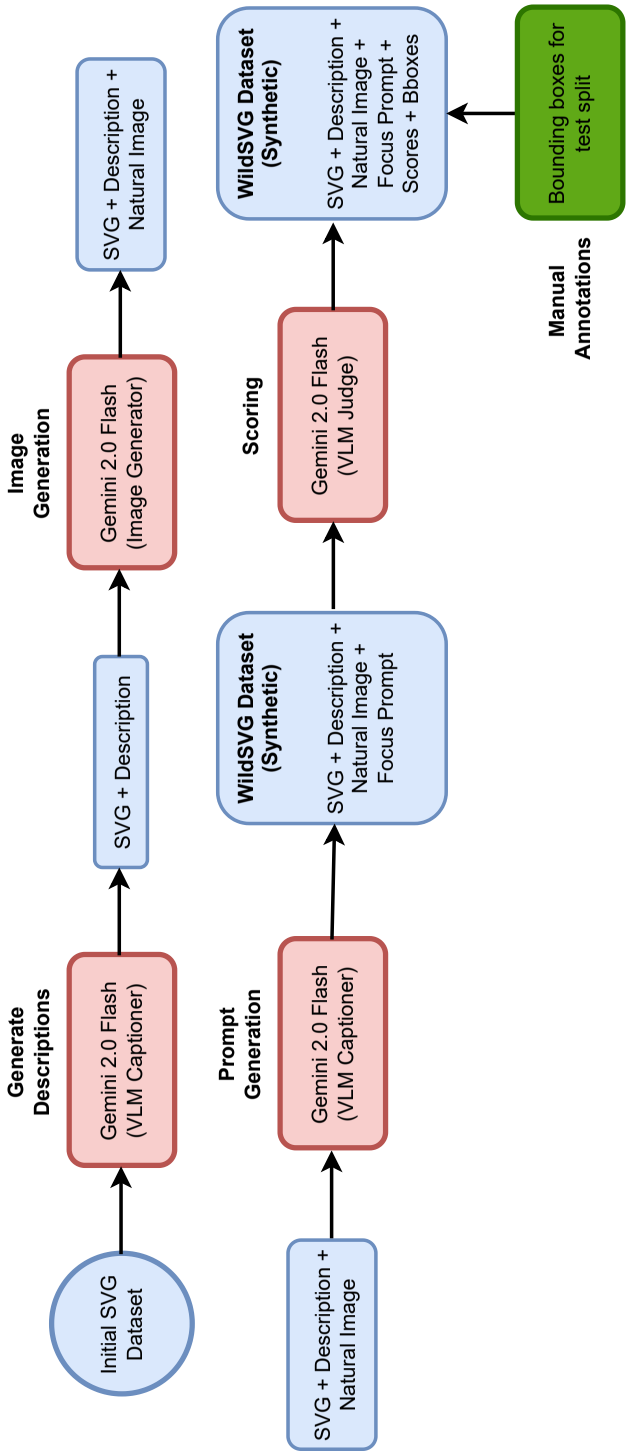

Figure 7: Pipeline for natural WildSVG generation

A.3.2 DATASET GENERATION PROMPTS

Prompt for Synthetic Image Generation

Integrate the image into a **real life photography** to help increase **difficulty** on a **dataset**. It must be **seamless** and the **integrated image** in the real life picture should **respect** the following **description**: {description}. First, decide on a **scenario** where to apply this SVG, it must be **semantically consistent**. An example could be a **publicity panel** or a **traffic sign** or a logo in a laptop, it is highly **encouraged** to think about **others scenarios**.

Figure 8: Prompt for image generation of Synthetic WildSVG Dataset

Prompt for VLLM scoring (Natural dataset)

You are a strict impartial evaluator of integrating SVG images into real life scenarios.
———— **RUBRIC** ————
Constancy Score (0-5) — How similar is the integrated image, does it maintain the original characteristics or does it remove them or add new things, without taking into account perspective or scale changes."
0 — **Completely unrecognizable**: Completely changed image.
1 — **Very weak recognition**: Some minor features are present but lack in key characteristics which makes it barely recognizable.
2 — **Weak recognition**: Some primary characteristics are presents but other key features are missing or completely changed. Very difficult to recognize as the integrated pictures.
3 — **Partial recognition**: Most important features are present, but some minor details or more secondary characteristics are missing or have being noticeable altered or newly added.
4 — **Strong recognition**: Recognizable, only some slight changes or addition in minor details have been done.
5 — **Perfect recognition**: Image is fully integrated with every minor detail conserved.

Alignment Score (0-5) — "How well adjusted is the information of a task prompt about the location of the integrated picture and what to extract."
0 — **Unusable**: Completely wrong location or completely wrong information about what to extract.
1 — **Very poor**: Difficult to understand or very ambiguous on what should be extracted.
2 — **Poor**: Somehow correct but confusing or ambiguous.
3 — **Fair**: Basic information without any class of details or additional information to deal with ambiguity.
4 — **Good**: Clear location and what to extract although some details are missing, leaving the possibility of some ambiguity.
5 — **Excellent**: Excellent information, clear and distinct leaving no room for ambiguity on what to extract.

———— **TASK** ————
Given the two images, one integrated into the other, and a task prompt with location information evaluate using the rubric; return the following JSON:
{"**constancy_score**": <integer 0-5>,
"**alignment_score**": <integer 0-5>,
"**justification**": 100-word explanation for each score}

Figure 9: Prompt for scoring Natural WildSVG instances

Prompt for VLLM scoring (Synthetic dataset)

You are a strict impartial evaluator of integrating SVG images into real life scenarios.
———— **RUBRIC** ————
Constancy Score (0-5) — How similar is the integrated image, does it maintain the original characteristics or does it remove them or add new things, without taking into account perspective or scale changes."
0 — **Completely unrecognizable**: Completely changed image.
1 — **Very weak recognition**: Some minor features are present but lack in key characteristics which makes it barely recognizable.
2 — **Weak recognition**: Some primary characteristics are presents but other key features are missing or completely changed. Very difficult to recognize as the integrated pictures.
3 — **Partial recognition**: Most important features are present, but some minor details or more secondary characteristics are missing or have being noticeable altered or newly added.
4 — **Strong recognition**: Recognizable, only some slight changes or addition in minor details have been done.
5 — **Perfect recognition**: Image is fully integrated with every minor detail conserved.

Aesthetics Score (0-5) — "Overall visual quality of the integration, if the integrated image has been synthetically inserted into the real life scenario."
0 — **Unusable**: The image has been copy and pasted directly onto a real life photo without any attention to perspective, illumination or basic coherence.
1 — **Very poor**: Despite some minor detail to give a more natural insertion, the image is still clearly inserted.
2 — **Poor**: Despite the details some key problems give away the synthetic insertion of the images.
3 — **Fair**: Insertion clear at first glance; acceptable composition; good enough to make doubt but has signs of synthetic insertion after an detailed observation.
4 — **Good**: Polished with only subtle imperfections, some minor illumination or perspective errors which make it difficult to discern if it was modified.
5 — **Excellent**: impossible to discern if the image is original or was modified.

Alignment Score (0-5) — "How well adjusted is the information of a task prompt about the location of the integrated picture and what to extract."
0 — Unusable: Completely wrong location or completely wrong information about what to extract.
1 — **Very poor**: Difficult to understand or very ambiguous on what should be extracted.
2 — **Poor**: Somehow correct but confusing or ambiguous.
3 — **Fair**: Basic information without any class of details or additional information to deal with ambiguity.
4 — **Good**: Clear location and what to extract although some details are missing, leaving the possibility of some ambiguity.
5 — **Excellent**: Excellent information, clear and distinct leaving no room for ambiguity on what to extract.

———— **TASK** ————
Given the two images, one integrated into the other, and a task prompt with location information evaluate using the rubric; return the following JSON:
{"**constancy_score**": <integer 0-5>,
"**aesthetics_score**": <integer 0-5>,
"**alignment_score**": <integer 0-5>,
"**justification**": 100-word explanation for each score}

Figure 10: Prompt for scoring Synthetic WildSVG instances

A.3.3 FILTERING SCORE FORMULAS

The filtering procedure was performed using the following equations1.

$$\begin{aligned} Synthetic\ Dataset\ Score &= 40\% \cdot C_{\text{score}} + 40\% \cdot AE_{\text{score}} + 20\% \cdot AL_{\text{score}}, \\ Natural\ Dataset\ Score &= 60\% \cdot C_{\text{score}} + 40\% \cdot AL_{\text{score}}. \end{aligned} \tag{1}$$

A.3.4 WILDSVG DATASET

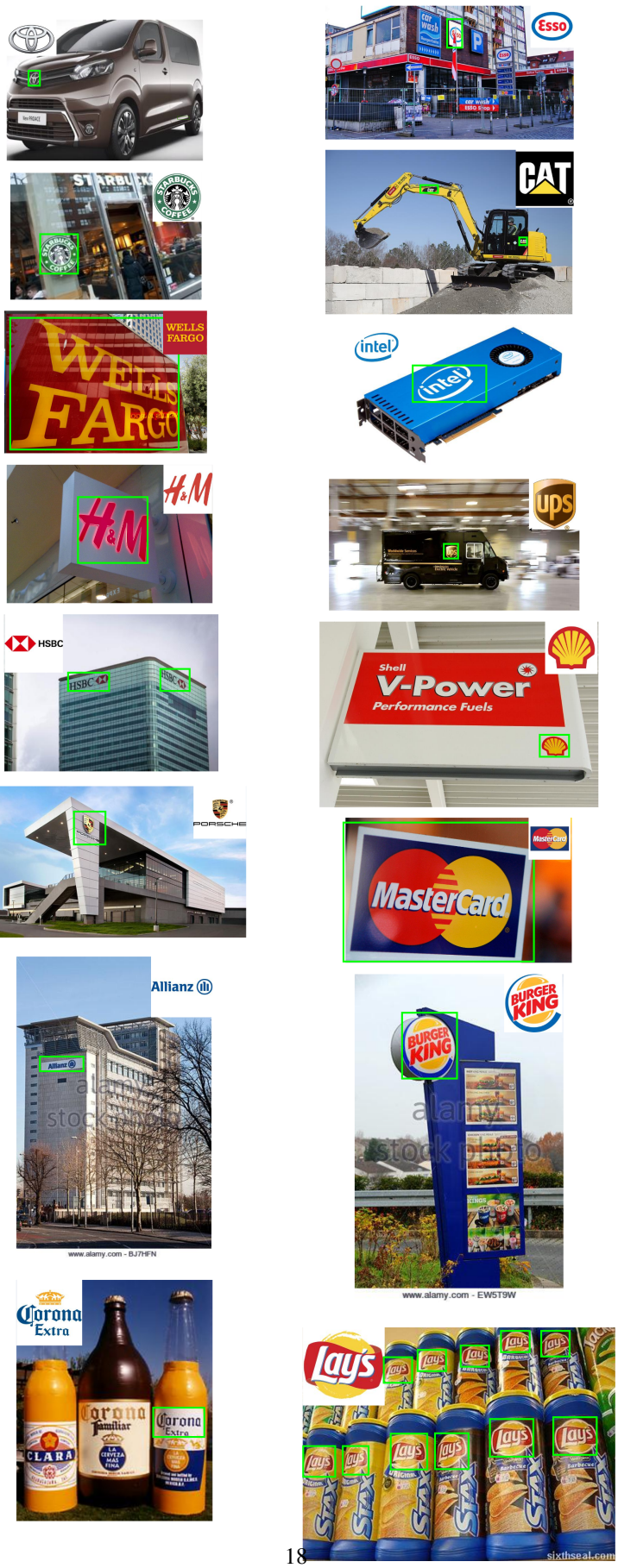

Figure 11: Examples of real-life images and associated SVG for natural WildSVG

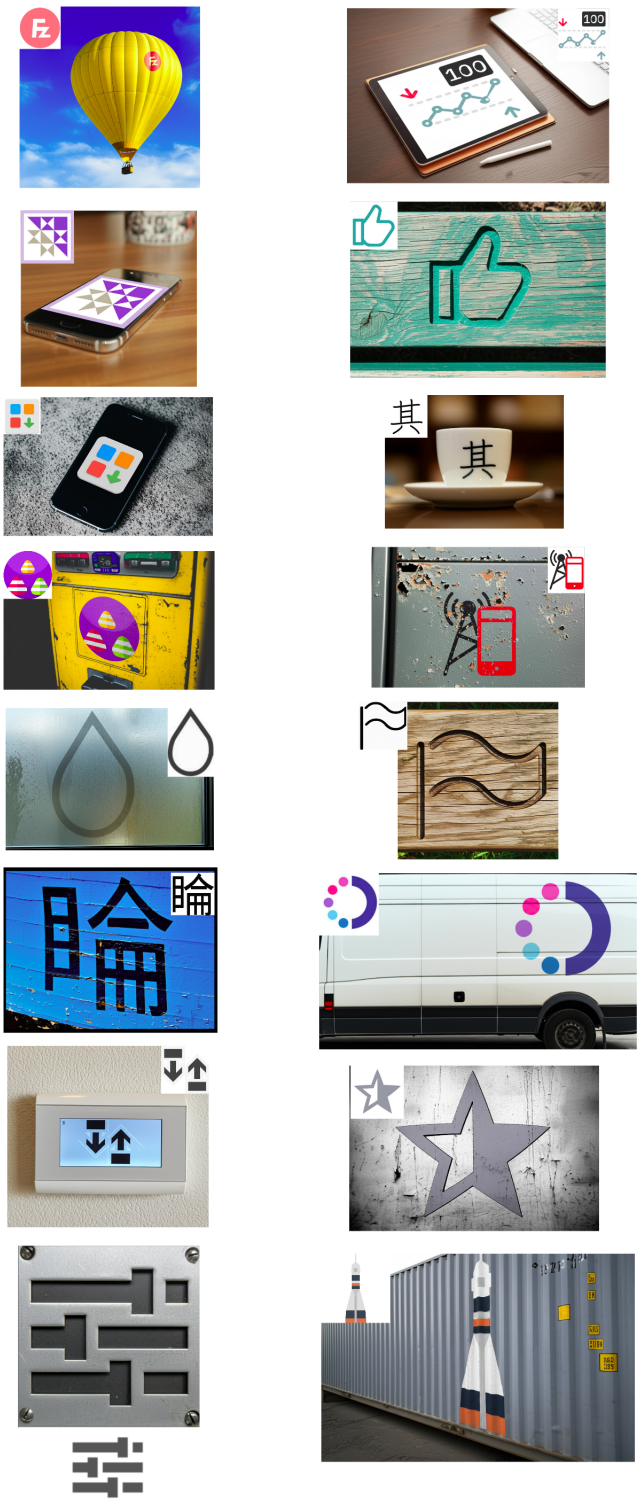

Figure 12: Examples of real-life images and associated SVG for synthetic WildSVG