StarVector: Generating Scalable Vector Graphics Code from Images and Text

Juan A. Rodriguez[1,2,4], **Abhay Puri**[1], **Shubham Agarwal**[1,2] **Issam H. Laradji**[1] **Sai Rajeswar**[1,2]
David Vazquez[1] **Christopher Pal**[1,2,3] **Marco Pedersoli**[1,4]

[1]ServiceNow Research
[2]Mila - Quebec AI Institute
[3]Canada CIFAR AI Chair
[4]École de Technologie Supérieure

Abstract

Scalable Vector Graphics (SVG) have become integral to modern image rendering applications due to their infinite scalability and versatility, especially in graphic design and web development. SVGs are essentially long strings of code that adhere to a structured syntax with validity constraints. With the rise of large language models, which excel at generating code in various languages, we aim to generate SVG code in a similar way. Our findings show that a vision-language model can be conditioned to produce valid SVG code that closely resembles input images, effectively enabling vectorization. Additionally, we harness the rich SVG syntax, encompassing all possible primitives—such as lines, paths, polygons, text, and effects like color gradients—that previous methods often missed. We briefly explain how the StarVector model operates, primarily leveraging a vision-language transformer architecture to generate SVG code. We also detail our training and inference procedures. Finally, we provide an interactive demo that allows users to input an image and generate its SVG code autoregressively, featuring real-time rendering that visually demonstrates the SVG generation process.

Code — https://starvector.github.io/

Introduction

Vector graphics provide an alternative to pixel-based images by maintaining image quality at any resolution(Li et al. 2020). This is because vector graphics are mathematically defined, and precisely parameterized to form shapes (Ma et al. 2022; Jain, Xie, and Abbeel 2023). They are widely used in graphic design and website development to display scalable, high-resolution visuals. Scalable Vector Graphics (SVG) (Quint 2003) is a popular format for vector images, offering a rich syntax for various shape elements.

With the rapid advancements in large language models for text generation (OpenAI 2023; Dubey et al. 2024) and their success in assisting with code generation (Nijkamp et al. 2022; Li et al. 2023b), applying these models to generate SVG code presents a promising opportunity. Our approach leverages the structured nature of SVG, utilizing a vision-language transformer architecture that can take images or text as input and directly output SVG code.

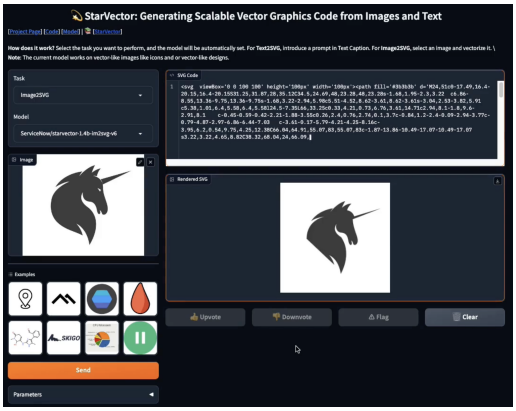

Figure 1: StarVector demo (SVG generation).

StarVector introduces a novel approach to SVG generation by framing it as a foundational image-to-SVG code generation task. We employ modern vision-language transformer architectures (Alayrac et al. 2022; Radford et al. 2021; Liu et al. 2023) to produce rich SVG code from images. In contrast, previous methods primarily focused on generating vector graphics relying on information in the image space using differentiable parameterizations, without explicitly modeling the underlying code (Lopes et al. 2019; Carlier et al. 2020; Reddy et al. 2021).

This paper overviews the StarVector method and describes how our demo works. The demo highlights how StarVector can accurately and efficiently generate SVG code that closely resembles an input image.

Related Work

SVG Generation. Early efforts in SVG generation relied on traditional image processing techniques (Li et al. 2020; Cortex 2023). With advancements in deep learning, methods like SVG-VAE (Lopes et al. 2019) and DeepSVG (Carlier et al. 2020) utilized latent variable methods for generating SVG, while Im2Vec (Reddy et al. 2021) decoded SVG from latent representations using recurrent networks. However, these approaches are limited to basic primitives like paths. More recent trends, such as VectorFusion (Jain, Xie, and Abbeel 2023), optimize SVGs with diffusion models,

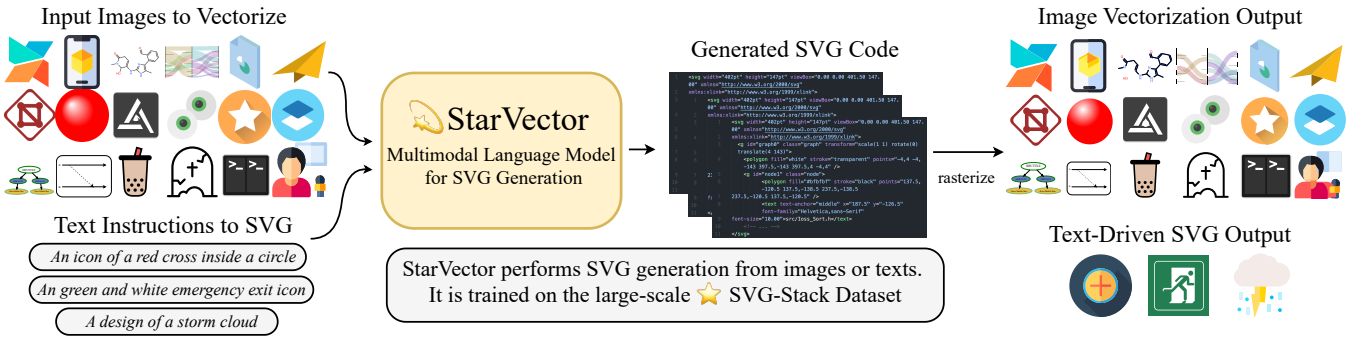

Figure 2: StarVector is a foundation model for SVG generation that uses a Vision-Language Modeling architecture to understand images and text. It generates direct SVG code autoregressively that can be rendered into logotypes, icons, or diagram images. (Left) raster images, (right) SVG images.

and IconShop (Wu et al. 2023) focuses on icon design restricted to paths. In contrast, our approach trains a foundation model that comprehends images and generates SVG code with complex syntax, including higher-order primitives like text rendering.

Language Models for Code Generation. Language models for code generation have gained popularity due to advances in text modeling (e.g. chatGPT) and the availability of large code datasets (Brown et al. 2020; OpenAI 2023; Husain et al. 2019). Models like StarCoder (Li et al. 2023b), and CodeLlama (Roziere et al. 2023) assist in generating and evaluating code across various languages. However, SVG code has largely been overlooked (Allal et al. 2023). Our work addresses this gap by designing and training a vision-language model specifically for SVG tasks, integrating image vectorization and text-conditioned SVG generation.

Multimodal Tasks and Models. Key vision-language tasks include image captioning (Li et al. 2022a,b) and text-to-image generation (Ramesh et al. 2021; Rombach et al. 2022). Leveraging techniques like ViT (Dosovitskiy et al. 2020), and CLIP (Radford et al. 2021), models like BLIP2 (Li et al. 2023a) and Llava (Liu et al. 2023) map image features into the LLM space, enabling free-form text generation. While previous works focus on reasoning over images (Radford et al. 2021), few explore converting images to code (Beltramelli 2018). We introduce the novel tasks of image-to-SVG, and text-to-SVG code generation.

StarVector Model Overview

As shown in Figure 3, StarVector consists of an image encoder and an LLM adapter that produce visual tokens, bridging images to the LLM space. Text instructions and SVG code are also tokenized and projected into this space. During training, visual and SVG code tokens are concatenated, and a standard language modeling objective is used to model the sequence that translates images into SVG codes, utilizing a transformer decoder with 1B parameters (Li et al. 2023b).

During inference, the encoder and adapter process the input image, and the model generates SVG code via autoregressive sampling. This approach enables StarVector to learn SVG syntax and effectively convert images into compilable SVG code, demonstrating its ability to draw using SVG code.

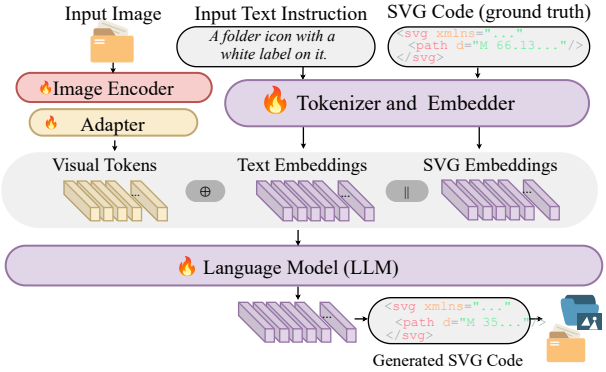

Figure 3: StarVector Architecture, Training, and Inference

SVG-Stack Training Dataset. A significant factor in StarVector's performance is its training on a large-scale corpus of image and SVG code pairs. A large dataset is essential for the model to learn SVG syntax and generalize to a wide range of complex SVG outputs. We constructed a large-scale dataset, **SVG-Stack**, that comprises over 2 million image-SVG code pairs, sourced from open and license-permissive repositories on GitHub. Specifically, we utilize TheStack (Kocetkov et al. 2022), a foundational code generation dataset, filtering out only the SVG code.

Demo Overview

The demo operates as follows: Users can either select an example image from the "Examples" section on the bottom left or upload an image, which will appear in the middle left area. For example, Figure 1 illustrates the process of uploading a unicorn image. After pressing "Send", the generation process begins, and the SVG code starts appearing in the upper right corner. Simultaneously, the SVG code is rendered in real-time below the SVG code, on the right side.

References

Alayrac, J.-B.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35: 23716–23736.

Allal, L. B.; Li, R.; Kocetkov, D.; Mou, C.; Akiki, C.; Ferrandis, C. M.; Muennighoff, N.; Mishra, M.; Gu, A.; Dey, M.; et al. 2023. SantaCoder: don't reach for the stars! *arXiv preprint arXiv:2301.03988*.

Beltramelli, T. 2018. pix2code: Generating code from a graphical user interface screenshot. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, 1–6.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Carlier, A.; Danelljan, M.; Alahi, A.; and Timofte, R. 2020. Deepsvg: A hierarchical generative network for vector graphics animation. *Advances in Neural Information Processing Systems*, 33: 16351–16361.

Cortex, V. 2023. VTracer. https://www.visioncortex.org/vtracer-docs. Accessed: 2023-11-01.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Husain, H.; Wu, H.-H.; Gazit, T.; Allamanis, M.; and Brockschmidt, M. 2019. Codesearchnet challenge: Evaluating the state of semantic code search. *arXiv preprint arXiv:1909.09436*.

Jain, A.; Xie, A.; and Abbeel, P. 2023. Vectorfusion: Text-to-svg by abstracting pixel-based diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1911–1920.

Kocetkov, D.; Li, R.; Allal, L. B.; Li, J.; Mou, C.; Ferrandis, C. M.; Jernite, Y.; Mitchell, M.; Hughes, S.; Wolf, T.; et al. 2022. The Stack: 3 TB of permissively licensed source code. *arXiv preprint arXiv:2211.15533*.

Li, C.; Xu, H.; Tian, J.; Wang, W.; Yan, M.; Bi, B.; Ye, J.; Chen, H.; Xu, G.; Cao, Z.; et al. 2022a. mplug: Effective and efficient vision-language learning by cross-modal skip-connections. *arXiv preprint arXiv:2205.12005*.

Li, J.; Li, D.; Savarese, S.; and Hoi, S. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*.

Li, J.; Li, D.; Xiong, C.; and Hoi, S. 2022b. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, 12888–12900. PMLR.

Li, R.; Allal, L. B.; Zi, Y.; Muennighoff, N.; Kocetkov, D.; Mou, C.; Marone, M.; Akiki, C.; Li, J.; Chim, J.; et al. 2023b. StarCoder: may the source be with you! *arXiv preprint arXiv:2305.06161*.

Li, T.-M.; Lukáč, M.; Gharbi, M.; and Ragan-Kelley, J. 2020. Differentiable vector graphics rasterization for editing and learning. *ACM Transactions on Graphics (TOG)*, 39(6): 1–15.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*.

Lopes, R. G.; Ha, D.; Eck, D.; and Shlens, J. 2019. A learned representation for scalable vector graphics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7930–7939.

Ma, X.; Zhou, Y.; Xu, X.; Sun, B.; Filev, V.; Orlov, N.; Fu, Y.; and Shi, H. 2022. Towards layer-wise image vectorization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16314–16323.

Nijkamp, E.; Pang, B.; Hayashi, H.; Tu, L.; Wang, H.; Zhou, Y.; Savarese, S.; and Xiong, C. 2022. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*.

OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774.

Quint, A. 2003. Scalable vector graphics. *IEEE MultiMedia*, 10(3): 99–102.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763. PMLR.

Ramesh, A.; Pavlov, M.; Goh, G.; Gray, S.; Voss, C.; Radford, A.; Chen, M.; and Sutskever, I. 2021. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, 8821–8831. PMLR.

Reddy, P.; Gharbi, M.; Lukac, M.; and Mitra, N. J. 2021. Im2Vec: Synthesizing Vector Graphics without Vector Supervision. *arXiv preprint arXiv:2102.02798*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Roziere, B.; Gehring, J.; Gloeckle, F.; Sootla, S.; Gat, I.; Tan, X. E.; Adi, Y.; Liu, J.; Remez, T.; Rapin, J.; et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Wu, R.; Su, W.; Ma, K.; and Liao, J. 2023. IconShop: Text-Based Vector Icon Synthesis with Autoregressive Transformers. *arXiv preprint arXiv:2304.14400*.