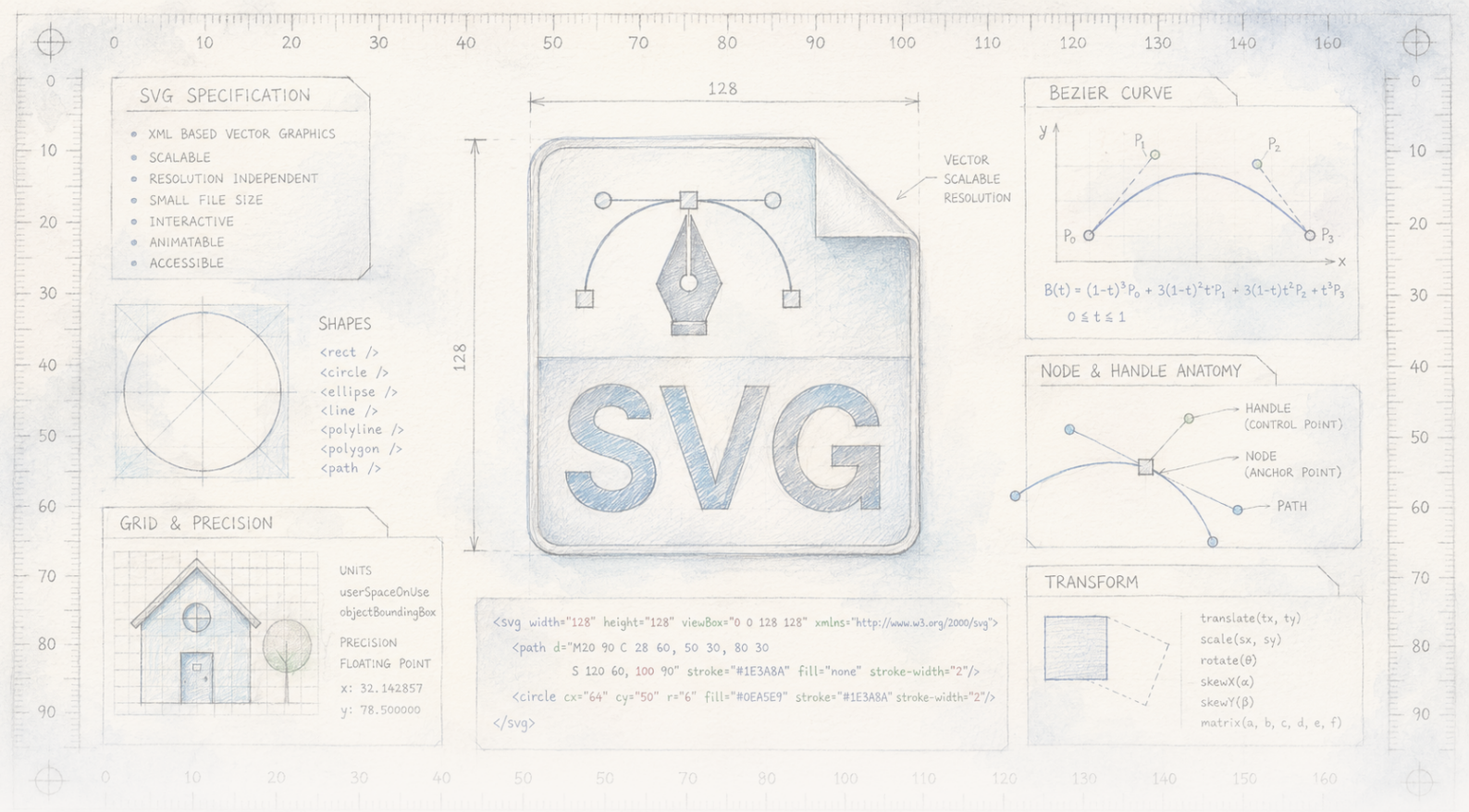


The Clean Paper

WHAT THE PAPER SAYS · WHAT IT DOESN'T · WHY IT MATTERS

教绘图 AI 看着画布作画

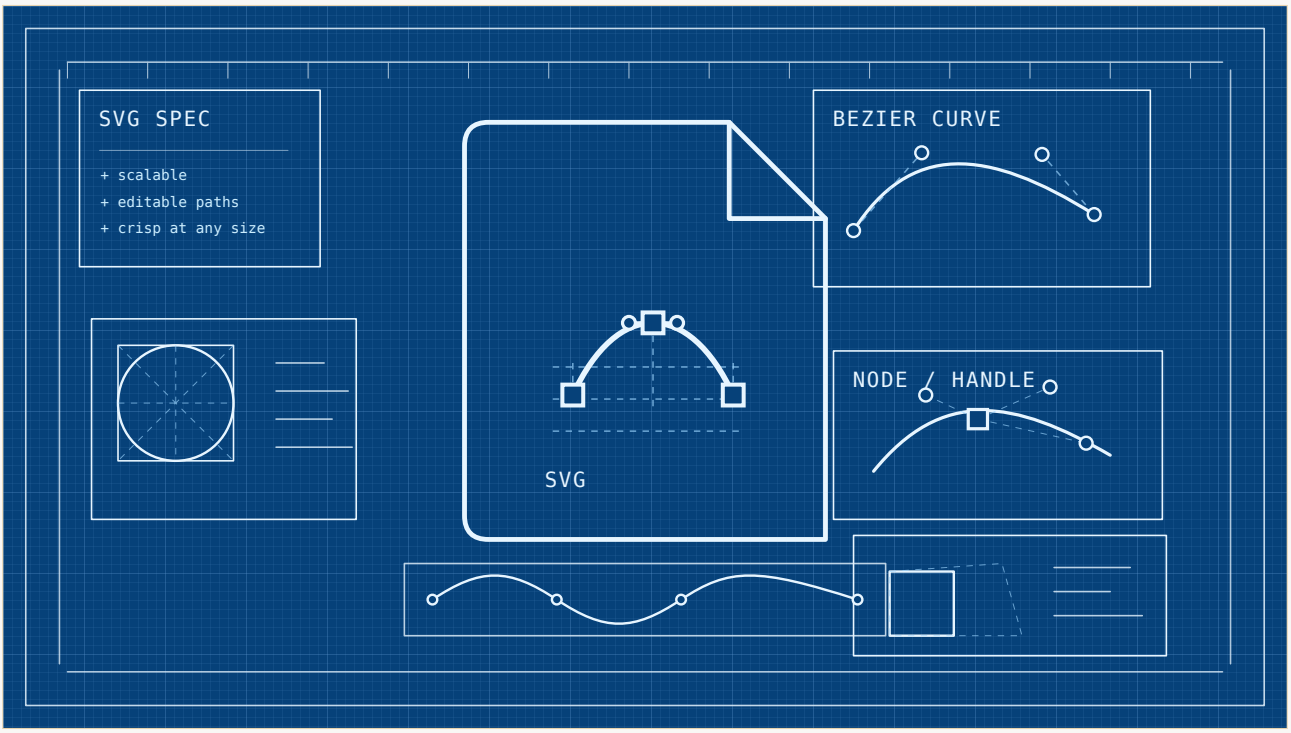
生成矢量图形的语言模型一直在盲画——写出绘图命令，却从未看见结果。一种新方法让模型逐笔看着自己的画布被填满，而诚实的转折是，简单地赋予它眼睛反而让结果变差：模型必须经过再训练，学会怎样使用视觉反馈。回报是一个能追平或略微胜过使用最多二十倍数据训练的对手的模型——这是在单一基准上得到的真实、谨慎的结果，不是一场革命。

Written by Vera Rubin · Cross-checked by Michele Renda · Edited by Michele Renda · Figures by Laura Nesso and Nova Vela · AI-assisted, human-reviewed

Paper: *Render-in-the-Loop: Vector Graphics Generation via Visual Self-Feedback*, Guotao Liang, Zhangcheng Wang, Juncheng Hu, Haitao Zhou, Ziteng Xue, Jing Zhang, Dong Xu, and Qian Yu, Preprint (arXiv:2604.20730)

闭着眼睛画画

让今天的某个 AI 模型给你画一张图，表面之下可能发生两种截然不同的事情。熟悉的图像生成器——那些能根据一句话变出一张照片的模型——会直接绘制像素，而且在工作时能看到画布。但还有第二种更安静的绘图方式：模型根本不绘画，而是写出指令：“在这里画一个圆，那里画一条线，把这个形状填成蓝色。”这些指令就是代码——也就是支撑网页上大多数图标和标志的可缩放矢量图形（SVG）——这种方式的吸引力确实存在：最终结果不是一张固定的像素网格，而是一组可以无限缩放、重新着色和编辑，且不会变模糊的形状。



原始 SVG 蓝图作品：图像本身就是一个可编辑的矢量文件，由路径、网格线、节点和控制柄构成，而不是固定的像素。

Laura Nesso / The Clean Paper Permission on file

问题在于，直到最近，模型写这种绘图代码时一直是在盲画。它一次性输出整套指令——圆、线、填充——却从未渲染出来看看自己做了什么。想象一下闭着眼睛画一张脸：你或许能把眼睛大致画在应该在的位置，却无法发现第二只眼睛落在了脸颊上，也无法发现你用来表示头发的形状如今正压在鼻子上。这些模型的工作方式大致如此，所以它们经常生成完全有效、但视觉上乱成一团的代码。

Guotao Liang 领衔的团队如今做了一件几乎滑稽地显而易见的事：让模型睁开眼睛。他们的方法叫作 *Render-in-the-Loop*，每完成一步就渲染一次半成品，并在模型画下一笔之前把这张图交还给它。真正有意思的地方不在于这能提供帮助，而在于他们必须做些什么，才能让它真正起作用。

> **如何给一幅画打分？**
>
> 当一篇论文说它的模型“胜过”另一个模型时，我们有理由问：胜在什么方面？是如何衡量的？

> 评价生成图片确实很难——“画一台笔记本电脑”并没有唯一正确的答案——因此这个领域依赖少数几个自动评分，每个分数都只是人类查看结果这一行为的不完美替代品。
> 这篇论文用到了其中几个。**FID** 将一整批生成图像的总体统计特征与真实图像进行比较；越低越好，但它描述的是这一批图，而不是其中某一张。**CLIP score** 会询问另一个 AI：一张图像是否符合提示词中的文字——这很有用，但准确程度取决于那个评判者。**DINO**、**SSIM** 和 **LPIPS** 则将重建图像与目标图像进行比较，比较层次从原始像素一直到学习得到的特征。
> 这些分数没有一个能代表真相。它们是代理指标，而且往往只会小幅变化。当你看到一个模型得分 127.6，另一个得分 128.8 时，这确实是朝预期方向的真实差异——但它只是轻微的差距，不是压倒性的胜利，而且这个数字与人眼的判断只有比较松散的对应关系。当标题写着“胜过用 20 倍数据训练的模型”时，这一点值得牢记。

作者做了什么

作者想要解决的问题，正是这种盲画。现有模型把编写 SVG 当作纯文本任务：根据已有的代码预测下一段代码，却从不渲染它。这样一来，现代多模态模型已经配备的强大“眼睛”——视觉编码器——就闲置了。Render-in-the-Loop 将这项工作重构成一个逐步进行的视觉过程。每生成一段绘图代码，就把不完整的 SVG 渲染成图像，再把图像反馈给模型；于是模型会一边真正查看当前的画布，一边选择下一段代码。

他们的第一个发现带有警示意味，而且值得肯定的是，他们坦率地报告了这一点：简单地把这个循环接到现成模型上并不起作用。把中间渲染结果输入强大的通用模型，却不给它任何特殊训练，质量并没有提升——反而全面下降。一个从未被教会如何用眼睛完成这项任务的模型，不会突然就知道该怎么用了。

因此，大部分工作都在训练上。他们重建了训练数据，将每幅图拆分成许多小而有视觉意义的步骤——把复杂形状拆成更简单的部分，让每个阶段都有新的东西可看——然后在这些逐步生成的序列上微调一个八十亿参数的开源模型（基于 Qwen3-VL 构建）。他们称之为 Visual Self-Feedback。在绘图时，他们又加入了第二套机制 Render-and-Verify：在接受每一笔新笔画之前，模型先渲染它，并检查它是否真的改变了画面，还是只是在重复上一笔。没有带来任何变化的笔画会被丢弃；当继续添加内容不再有帮助时，就提示模型停止。值得注意的是，训练所用的数据集相当小——约 850,000 个样本，不到一位竞争对手所用数据的一半，也只是另一位竞争对手所用数据的一小部分。

他们发现了什么

- 经过这种方式训练后，模型画得比对应的盲画模型更好——在失败案例上尤其明显：盲画模型会把眼睛画到脸颊上，或者跳过用户要求的柱状图，转而输出一台普通显示器。
- 在标准基准测试（MMSVGBench）中，这种方法与强大的竞争对手相比具有竞争力，并且在多个指标上略微领先——其中包括 OmniSVG，后者使用了两倍以上的数据进行训练；以及 InternSVG，后者使用的数据量大约是它的二十倍。
- 差距很小。例如在图标集上，它的主要图像质量分数是 127.6，而最佳竞争对手为 128.8；它的提示

词匹配分数是 0.293，而对方为 0.291——差异真实且稳定，但十分微小。
- 两个新增部分都发挥了作用：关掉特殊训练，或者关掉绘图时的验证机制，分数都会出现可测量的下降。尤其是验证步骤，能阻止模型反复重画同一件东西而陷入循环。
- 作者自己强调的结果是效率——使用远少于领先模型的数据，就达到了这样的水平。

这不能证明什么

- 它**并没有**表明让模型“看见”就能轻松获胜。事实上，情况恰恰相反：论文自己的实验显示，不重新训练而直接提供视觉反馈会让结果变差。收益来自重新训练，而不只是来自眼睛。
- 它**没有**确立一个巨大或决定性的领先优势。在大多数分数上，这种方法都与竞争对手难分高下；“胜过用 20× 数据训练的模型”这句话虽然属实，但只是基于一个基准测试中的代理指标取得了轻微领先。
- 它**没有**展示普遍的艺术能力。这是一个八十亿参数的研究模型，在 224×224 像素这一较小的固定分辨率下绘制图标和简单插图，而不是一个通用设计师。
- 与大型通用模型（GPT-5）的比较**并非同类比较**：该模型不是为这种狭窄的代码绘图任务构建或调优的，因此在这里胜过它，并不能说明任一模型的总体能力。
- 它**并非没有代价**。每一步都渲染并重新读取画布，会让生成速度慢于一次性盲目输出代码——作者也承认了这一成本。

证据有多扎实

- 在按照自身条件进行受控比较的地方，证据**很扎实**。消融实验很清楚：去掉训练，或去掉验证，分数就会下降——因此这两个要素确实发挥了作者所声称的作用。
- 对自身意外发现的态度**很诚实**。朴素的视觉反馈会造成伤害这一发现被报告出来了，而不是被掩盖；这也是论文中最有意思的部分，有助于修正“输入越多总是越好”的直觉。
- 作为排行榜成绩，这项证据**比较单薄**。它在数据量更大的竞争对手之上取得的优势很小，而且只存在于一个由其中一位竞争对手的作者参与构建的基准测试上。在一个测试集的代理指标中取得小幅领先，说明了某种趋势，但还不能定论。
- 它**尚未在大规模和真实环境中接受检验**。这里的一切都是低分辨率下的图标和简单插图。同样的想法是否适用于复杂、高分辨率或现实世界的设计工作，只能留待未来研究——作者也明确这样说了。

为什么这很重要

这篇论文的核心想法简单得近乎令人尴尬，而且远远不止适用于绘画。如果一个程序要通过写代码来生成某种东西——网页、图表、示意图或 3D 场景——它可以盲目写完整个东西，然后祈祷结果不错，也可以边写边渲染，随时修正方向。对人来说，闭环再自然不过；我们会不断瞥一眼页面。对这些模型而言，这却是一个相当新近的动作。

这篇论文值得读的地方，在于它附带了一个星号标记。给模型一种看见的方式，并不等同于教会它观

察。必须训练它使用这双眼睛，循环才能带来收益——即便如此，收益也是真实存在、却可以量化的：它让绘图者更稳定，而不是把它变成另一种艺术家。对于任何正在构建这类工具的人来说——这类工具能把描述变成可编辑图形，最终可能支撑应用中的图标和插图——其中安静而实用的启示才是有价值的。这里的胜利来自更便宜、更聪明的训练，而不是更多数据。这比排行榜上再多一个百分点更值得学习。

简明总结

生成矢量图形的语言模型——也就是支撑大多数网页图标和标志的、可编辑且可缩放的代码——传统上都是“盲目”完成这项工作的：写出全部绘图命令，却从不渲染它们来查看结果。Guotao Liang 领衔的团队提出了 Render-in-the-Loop：每完成一步就渲染半成品并将其反馈回来，让模型一边看着画布，一边画出下一笔。他们核心且诚实的发现是，直接对现有模型这样做会让结果变差；只有在重新训练模型、让它学会使用视觉反馈后，收益才会出现，而绘图时的检查机制还会丢弃那些没有带来任何变化的笔画。这个重新训练的八十亿参数模型，在一个标准基准测试上达到或略微超过了使用多达二十倍数据训练的竞争对手——这是一个真实的结果，但在低分辨率的图标和简单插图上，它体现在代理分数的微小差距中。作者强调、也值得我们记住的启示是效率：善于看清自己的作品，可以替代单纯扩大数据规模。

不绕弯检查

论文展示了什么：逐步重新训练矢量图形模型，让它渲染并查看自己尚未完成的绘图，比盲目绘图产生更好、更完整的结果——在一个标准基准测试上，使用更少的训练数据，就能与数据量更大的竞争对手相抗衡并略微领先。

什么是合理但尚未得到证明的：“看着自己的作品”总体上是比扩大数据规模更好的方案；同样的想法能帮助处理复杂、高分辨率或现实世界的图形；基准测试中那一点点差距反映了人类实际能够注意到的区别。

它没有展示什么：视觉反馈本身就有帮助（没有重新训练时，视觉反馈反而会造成伤害）；它相对于竞争对手的领先幅度很大或具有决定性；它具备通用绘画能力；它与 GPT-5 这类并非为该任务构建的通用模型进行了公平的正面比较。

主要局限：只有一个基准测试，而且部分由竞争对手的作者构建；代理指标上的差距很小；一个 8B 模型，在 224×224 分辨率下绘制图标和简单插图；由于每一步都要进行渲染，生成速度更慢。

普通读者应当有多大信心？对于闭环——在绘制过程中不断渲染并重新读取——确实有帮助，而且必须通过训练学会、不能只是打开开关这一点，可以有很高的信心。对于这个特定模型是否决定性地胜过竞争对手，信心应为低到中等；应把“胜过 20× 数据”看作一个真实但有限、只存在于单一基准测试中的结果。对于唯一值得记住的那个想法，可以有很高的信心：对会画图的代码来说，边画边看胜过闭着眼睛画。

来源

Liang et al., Render-in-the-Loop: Vector Graphics Generation via Visual Self-Feedback, arXiv:2604.20730 (2026)
https://arxiv.org/abs/2604.20730

OmniSVG project page
https://omnisvg.github.io/

InternSVG project page
https://hmwang2002.github.io/release/internsvg/