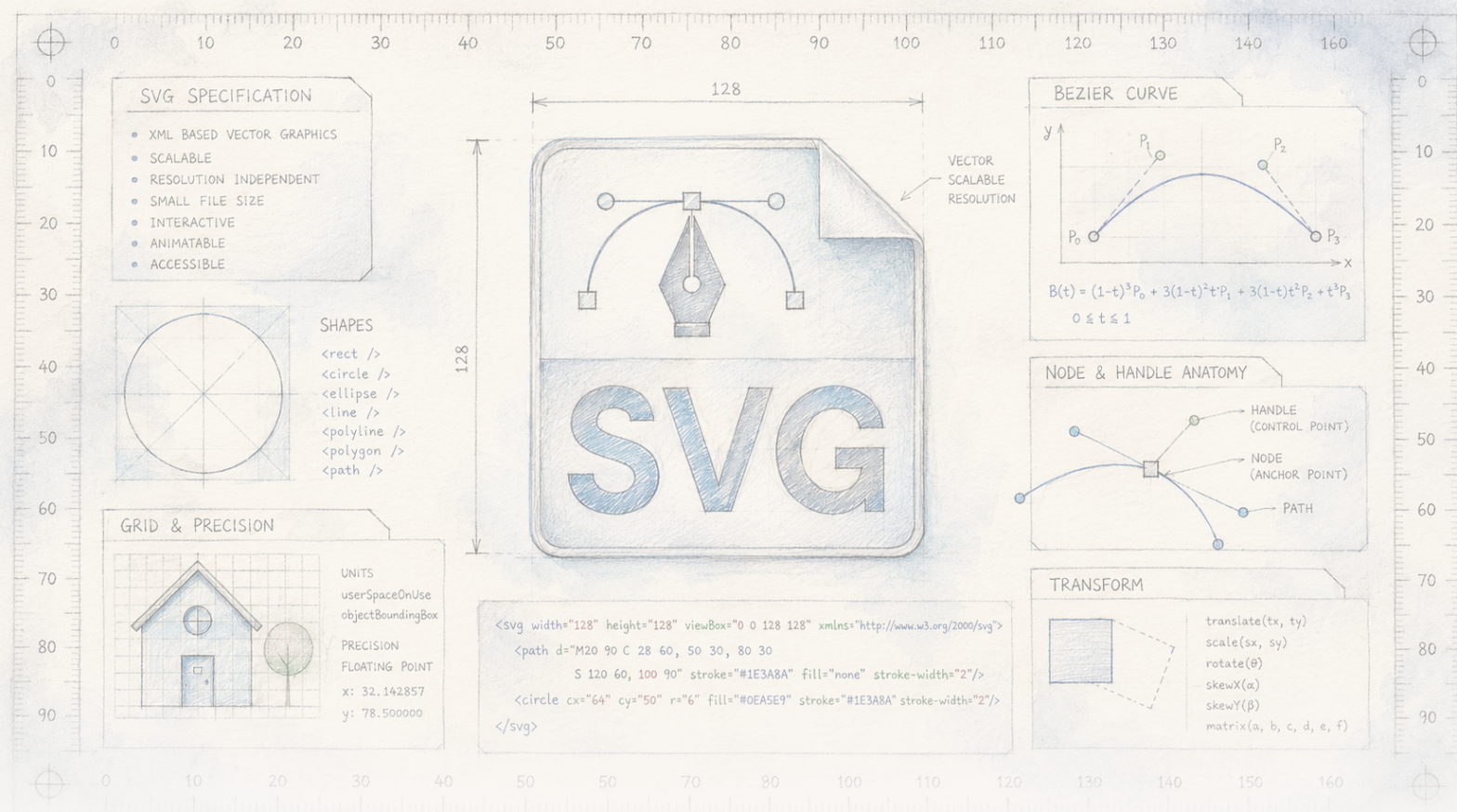




The CLEAN PAPER

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

Mengajari AI penggambar untuk melihat kanvasnya

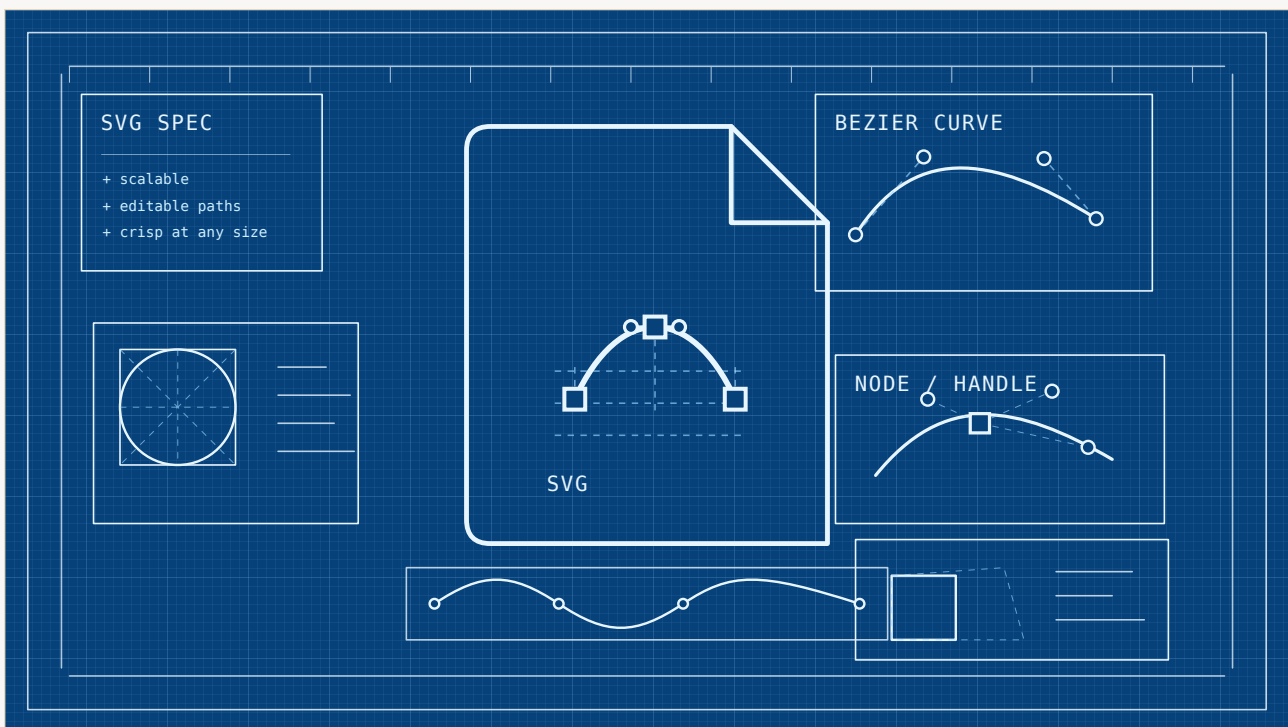
Language model yang menghasilkan grafik vektor bekerja secara buta — menulis perintah gambar tanpa pernah melihat hasilnya. Metode baru membuat model melihat kanvasnya sendiri terisi, goresan demi goresan, dan bagian jujurnya adalah bahwa sekadar memberinya “mata” justru membuat hasil lebih buruk: model harus dilatih ulang untuk menggunakannya. Hasil akhirnya adalah model yang menyamai atau sedikit melampaui pesaing yang dilatih dengan hingga dua puluh kali lebih banyak data — hasil nyata dan hati-hati pada satu benchmark, bukan revolusi.

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)

Menggambar dengan mata tertutup

Minta salah satu model AI masa kini menggambar sesuatu, dan di balik layar salah satu dari dua hal yang sangat berbeda akan terjadi. Generator gambar yang sudah akrab — yang dapat memunculkan foto dari satu kalimat — melukis piksel secara langsung, dan dapat “melihat” kanvas saat bekerja. Namun ada jenis menggambar kedua yang lebih sunyi, ketika model sama sekali tidak melukis. Ia menulis instruksi: *gambar lingkaran di sini, garis di sana, isi bentuk ini dengan warna biru*. Instruksi itu adalah kode — Scalable Vector Graphics atau SVG yang sama yang berada di balik sebagian besar ikon dan logo web — dan daya tariknya nyata: hasilnya bukan kisi piksel tetap, melainkan kumpulan bentuk yang dapat diperbesar, diwarnai ulang, dan diedit terus-menerus tanpa menjadi buram.



Karya blueprint SVG asli: gambarnya sendiri adalah file vektor yang dapat diedit, dibangun dari path, garis kisi, node, dan handle, bukan piksel tetap.

Laura Nesso / The Clean Paper Permission on file

Masalahnya, hingga baru-baru ini model yang menulis kode gambar seperti itu bekerja secara buta. Ia mengeluarkan seluruh rangkaian instruksi dalam sekali jalan — lingkaran, garis, isi — tanpa pernah merendernya untuk melihat apa yang telah dibuat. Bayangkan menggambar wajah dengan mata tertutup: kita mungkin menempatkan mata kira-kira di lokasi yang benar, tetapi tidak akan tahu kalau mata kedua jatuh di pipi, atau bentuk rambut ternyata menutupi hidung. Kurang lebih begitulah model-model ini bekerja, sehingga mereka sering menghasilkan kode yang sepenuhnya valid tetapi berantakan secara visual.

Tim yang dipimpin Guotao Liang kini melakukan hal yang nyaris terlalu jelas: mereka membiarkan model membuka mata. Metode mereka, *Render-in-the-Loop*, merender gambar yang belum selesai setelah setiap langkah dan mengembalikan gambar itu kepada model sebelum ia membuat goresan

berikutnya. Bagian yang benar-benar menarik bukan bahwa cara ini membantu — melainkan apa yang harus mereka lakukan agar bantuan itu muncul sama sekali.

Bagaimana cara memberi nilai pada sebuah gambar?

Ketika makalah mengatakan modelnya “mengalahkan” model lain, wajar bertanya: mengalahkan dalam hal apa, dan diukur bagaimana? Menilai gambar hasil generasi memang sulit — tidak ada satu jawaban benar untuk “gambar sebuah laptop” — sehingga bidang ini bergantung pada beberapa skor otomatis, masing-masing hanya pengganti tidak sempurna bagi manusia yang melihat hasilnya.

Beberapa muncul di makalah ini. **FID** membandingkan karakter statistik keseluruhan dari sekumpulan gambar generasi dengan gambar nyata; lebih rendah lebih baik, tetapi ia menggambarkan satu batch, bukan satu gambar tertentu. **CLIP score** meminta AI lain menilai apakah gambar cocok dengan kata-kata prompt — berguna, tetapi hanya sebaik penilainya. **DINO**, **SSIM**, dan **LPIPS** membandingkan gambar rekonstruksi dengan target, mulai dari tingkat piksel mentah sampai fitur yang dipelajari.

Tidak satu pun adalah kebenaran. Semuanya proxy, dan biasanya bergerak dalam selisih kecil. Jika satu model mendapat 127,6 sedangkan yang lain 128,8, itu memang perbedaan nyata ke arah yang diinginkan — tetapi hanya dorongan kecil, bukan kemenangan telak, dan terjadi pada angka yang hanya secara longgar mengikuti penilaian mata manusia. Hal itu perlu diingat ketika judulnya berbunyi “mengalahkan model yang dilatih dengan data dua puluh kali lebih banyak”.

Apa yang dilakukan para penulis

Masalah yang hendak diperbaiki para penulis adalah proses menggambar buta itu sendiri. Model yang ada memperlakukan penulisan SVG sebagai tugas teks murni: prediksi potongan kode berikutnya dari kode yang sudah ada, tanpa pernah merendernya. Akibatnya “mata” yang kuat — vision encoder — yang sudah dimiliki model multimodal modern dibiarkan menganggur. Render-in-the-Loop menyusun ulang pekerjaan ini menjadi proses visual langkah demi langkah. Setelah setiap fragmen kode gambar, SVG parsial dirender menjadi gambar dan dikembalikan kepada model, sehingga fragmen berikutnya dipilih sambil benar-benar melihat kanvas yang sudah terbentuk.

Temuan pertama mereka justru peringatan, dan patut dipuji karena dilaporkan dengan jelas: sekadar menempelkan loop ini pada model yang sudah ada tidak bekerja. Ketika render antara diberikan kepada model umum yang kuat tanpa pelatihan khusus, kualitas tidak membaik — justru *menurun* secara menyeluruh. Model yang tidak pernah diajari memakai matanya untuk tugas ini tidak tiba-tiba tahu caranya.

Jadi sebagian besar pekerjaannya berada pada pelatihan. Mereka menyusun ulang data latih agar setiap gambar dipecah menjadi banyak langkah kecil yang bermakna secara visual — membagi bentuk kompleks menjadi bagian sederhana agar selalu ada sesuatu yang baru untuk dilihat pada tiap tahap — lalu melakukan fine-tuning pada model terbuka delapan miliar parameter berbasis Qwen3-

VL menggunakan urutan langkah tersebut. Mereka menyebutnya Visual Self-Feedback. Mekanisme kedua ditambahkan saat menggambar, Render-and-Verify: sebelum menerima setiap goresan baru, model merendernya dan memeriksa apakah goresan itu benar-benar mengubah gambar atau hanya mengulang yang sebelumnya. Goresan yang tidak menambah apa pun di buang, dan ketika tidak ada lagi yang membantu, model diminta berhenti. Menariknya, semua ini berjalan pada kumpulan data yang relatif kecil — sekitar 850.000 contoh, kurang dari setengah data yang dipakai salah satu pesaing dan hanya sebagian kecil dari pesaing lainnya.

Apa yang mereka temukan

- Setelah dilatih dengan cara ini, model menggambar lebih baik daripada versi butanya — paling jelas pada kasus gagal, ketika model buta menaruh mata di pipi, atau melewati diagram batang yang diminta dan malah menghasilkan monitor generik.
- Pada tolok ukur standar MMSVGBench, metode ini kompetitif dengan, dan pada beberapa ukuran sedikit melampaui, pesaing kuat — termasuk OmniSVG, yang dilatih dengan lebih dari dua kali lipat data, dan InternSVG, dengan sekitar dua puluh kali lipat data.
- Selisihnya kecil. Pada set ikon, misalnya, skor utama kualitas gambarnya 127,6 versus 128,8 pada pesaing terbaik, dan skor kecocokan prompt 0,293 versus 0,291 — nyata dan konsisten, tetapi tipis.
- Kedua tambahan memang berguna: matikan pelatihan khusus atau verifikasi saat menggambar, dan angkanya turun secara terukur. Langkah verifikasi terutama mencegah model terjebak menggambar ulang hal yang sama berulang-ulang.
- Hasil yang paling ditekankan para penulis adalah efisiensi — mencapai tingkat ini dengan data pelatihan jauh lebih sedikit daripada para pemimpin.

Apa yang *tidak* dibuktikan

- Studi ini **tidak** menunjukkan bahwa membiarkan model “melihat” otomatis menguntungkan. Justru sebaliknya: eksperimen mereka sendiri menunjukkan umpan balik visual *tanpa* pelatihan ulang membuat hasil lebih buruk. Keuntungan datang dari pelatihan ulang, bukan dari mata saja.
- Studi ini **tidak** menunjukkan keunggulan besar atau menentukan. Pada kebanyakan skor, metode ini berimbang dengan pesaing; “mengalahkan model yang dilatih dengan 20× data” memang benar, tetapi hanya dengan selisih kecil pada metrik proxy di satu tolok ukur.
- Studi ini **tidak** menunjukkan kemampuan seni umum. Ini model riset delapan miliar parameter yang menggambar ikon dan ilustrasi sederhana pada resolusi tetap kecil (224×224 piksel), bukan desainer serbaguna.
- Perbandingan dengan model umum besar (GPT-5) **bukan** perbandingan setara: model itu tidak dibangun atau di-tune untuk tugas sempit menggambar lewat kode ini, sehingga kemenangan di sini tidak banyak mengatakan tentang kedua model secara umum.
- Studi ini **tidak** gratis dari sisi komputasi. Merender dan membaca ulang kanvas pada setiap langkah membuat generasi lebih lambat daripada mengeluarkan seluruh kode dalam satu pass buta — biaya yang diakui penulis.

Seberapa kuat buktinya?

- **Kuat** untuk perbandingan terkontrol sesuai ruang lingkupnya. Ablation-nya bersih: hapus pelatihan atau verifikasi, dan angkanya turun — jadi kedua komponen benar-benar melakukan pekerjaan yang diklaim.
- **Jujur mengenai kejutan sendiri.** Temuan bahwa umpan balik visual naif justru *merugikan* dilaporkan, bukan disembunyikan, dan merupakan salah satu bagian paling menarik: koreksi berguna terhadap intuisi bahwa lebih banyak input selalu lebih baik.
- **Tipis untuk klaim papan peringkat.** Kemenangan atas pesaing dengan data lebih besar kecil dan berada pada satu tolok ukur yang dibuat oleh penulis salah satu pesaing tersebut. Selisih kecil pada metrik proxy di satu set pengujian adalah petunjuk, bukan putusan final.
- **Belum diuji pada skala dan dunia nyata.** Semua yang ada di sini berupa ikon dan ilustrasi sederhana beresolusi rendah. Apakah ide yang sama bekerja pada desain kompleks, resolusi tinggi, atau pekerjaan nyata masih menjadi pekerjaan masa depan — dan para penulis mengatakannya.

Mengapa ini penting

Gagasan di pusat makalah ini hampir terlalu sederhana, tetapi menjangkau jauh melampaui gambar. Jika sebuah program akan menghasilkan sesuatu dengan menulis kode — halaman web, chart, diagram, adegan 3D — ia dapat menulis semuanya secara buta lalu berharap, atau merender sambil berjalan dan mengoreksi arah. Menutup loop itu sudah menjadi kebiasaan alami bagi manusia; kita terus melirik halaman saat bekerja. Anehnya, bagi model-model ini langkah tersebut tergolong baru.

Yang membuat makalah ini layak dibaca adalah tanda bintang yang ditempelkannya pada ide tersebut. Memberi model cara untuk melihat tidak sama dengan mengajarnya memperhatikan. “Mata” itu harus dilatih, baru loop memberi hasil — dan bahkan setelah itu hasilnya nyata tetapi terukur: juru gambar yang lebih stabil, bukan jenis seniman yang sama sekali baru. Bagi siapa pun yang membangun alat yang mengubah deskripsi menjadi grafik yang dapat diedit — seperti ikon dan ilustrasi aplikasi — pelajaran praktis yang sunyi justru paling berguna. Kemenangan di sini berasal dari pelatihan yang lebih murah dan lebih cerdas, bukan sekadar data lebih banyak. Itu lebih berharga daripada satu poin lagi di papan peringkat.

Singkatnya

Language model yang menghasilkan grafik vektor — kode yang dapat diedit dan diskalakan di balik sebagian besar ikon dan logo web — secara tradisional melakukannya secara “buta”, menulis seluruh perintah gambar tanpa pernah merender dan melihat hasilnya. Tim yang dipimpin Guotao Liang mengusulkan Render-in-the-Loop: render gambar setengah jadi setelah setiap langkah lalu kembalikan kepada model, sehingga goresan berikutnya dibuat sambil melihat kanvas. Temuan jujur mereka yang utama adalah bahwa sekadar melakukan ini pada model yang ada justru membuatnya *lebih buruk*; keuntungan baru muncul setelah model dilatih ulang untuk memakai umpan balik visual,

dibantu pemeriksaan saat menggambar yang membuang goresan yang tidak mengubah apa pun. Model delapan miliar parameter yang dilatih ulang menyamai atau sedikit mengungguli pesaing yang dilatih dengan hingga dua puluh kali lebih banyak data pada satu tolok ukur standar — hasil nyata, tetapi dengan selisih kecil pada skor proxy, untuk ikon dan ilustrasi sederhana beresolusi rendah. Inti yang ditekankan penulis adalah efisiensi: melihat pekerjaan sendiri dengan baik dapat menggantikan sebagian skala data semata.

Penilaian tanpa berlebihan

Apa yang ditunjukkan makalah: Melatih ulang model grafik vektor agar merender dan melihat gambar setengah jadinya sendiri, langkah demi langkah, menghasilkan gambar yang lebih baik dan lebih lengkap daripada menggambar buta — kompetitif dengan dan sedikit di depan pesaing dengan data lebih besar pada satu tolok ukur standar, menggunakan data pelatihan lebih sedikit.

Apa yang masuk akal tetapi belum terbukti: Bahwa “melihat pekerjaan sendiri” secara umum merupakan resep lebih baik daripada sekadar menambah data; bahwa ide yang sama membantu grafik kompleks, resolusi tinggi, atau dunia nyata; bahwa selisih tolok ukur kecil cukup besar untuk benar-benar terlihat oleh manusia.

Apa yang tidak ditunjukkan: Bahwa umpan balik visual membantu dengan sendirinya — tanpa pelatihan ulang justru merugikan; bahwa keunggulan atas pesaing besar atau menentukan; kemampuan menggambar serbaguna; perbandingan setara dengan model umum seperti GPT-5 yang tidak dibangun untuk tugas ini.

Keterbatasan utama: Satu tolok ukur yang sebagian dibuat oleh penulis pesaing; selisih kecil pada metrik proxy; model 8B, ikon dan ilustrasi sederhana pada 224×224 ; generasi lebih lambat karena loop render di setiap langkah.

Seberapa besar keyakinan yang sebaiknya dimiliki pembaca umum? Tinggi bahwa menutup loop — merender dan membaca ulang sambil menggambar — benar-benar membantu, dan bahwa kemampuan itu harus dilatih, bukan sekadar dinyalakan. Rendah hingga sedang bahwa model khusus ini secara tegas mengalahkan para pesaing; perlakukan “mengalahkan $20 \times$ data” sebagai hasil nyata tetapi sederhana dari satu tolok ukur. Tinggi untuk satu ide yang layak diingat: bagi kode yang menggambar, melihat sambil bekerja lebih baik daripada menggambar buta.

Sumber

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://hwwang2002.github.io/release/internsvg/>