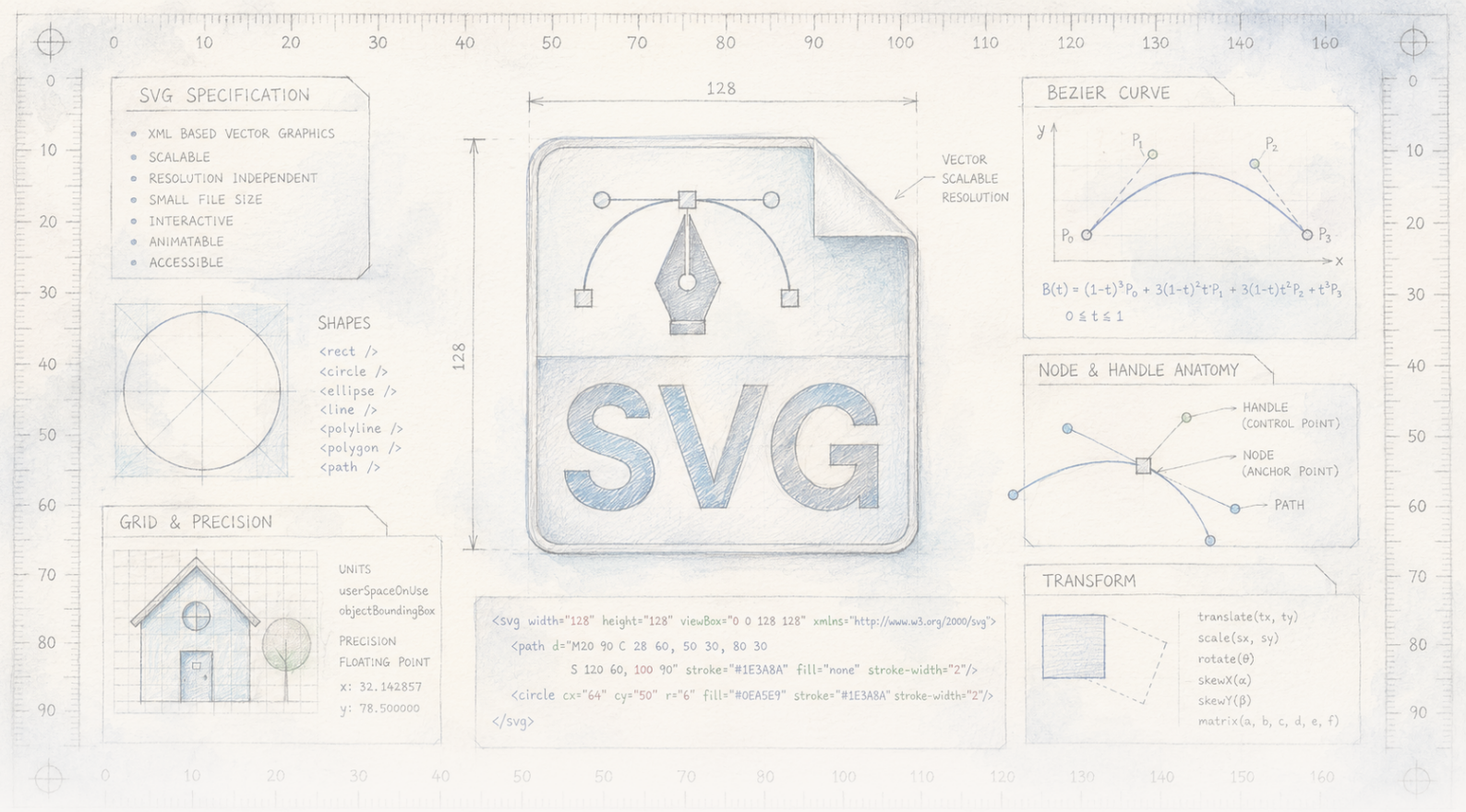


The Clean Paper

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

Да научиш AI што црта да гледа во страницата

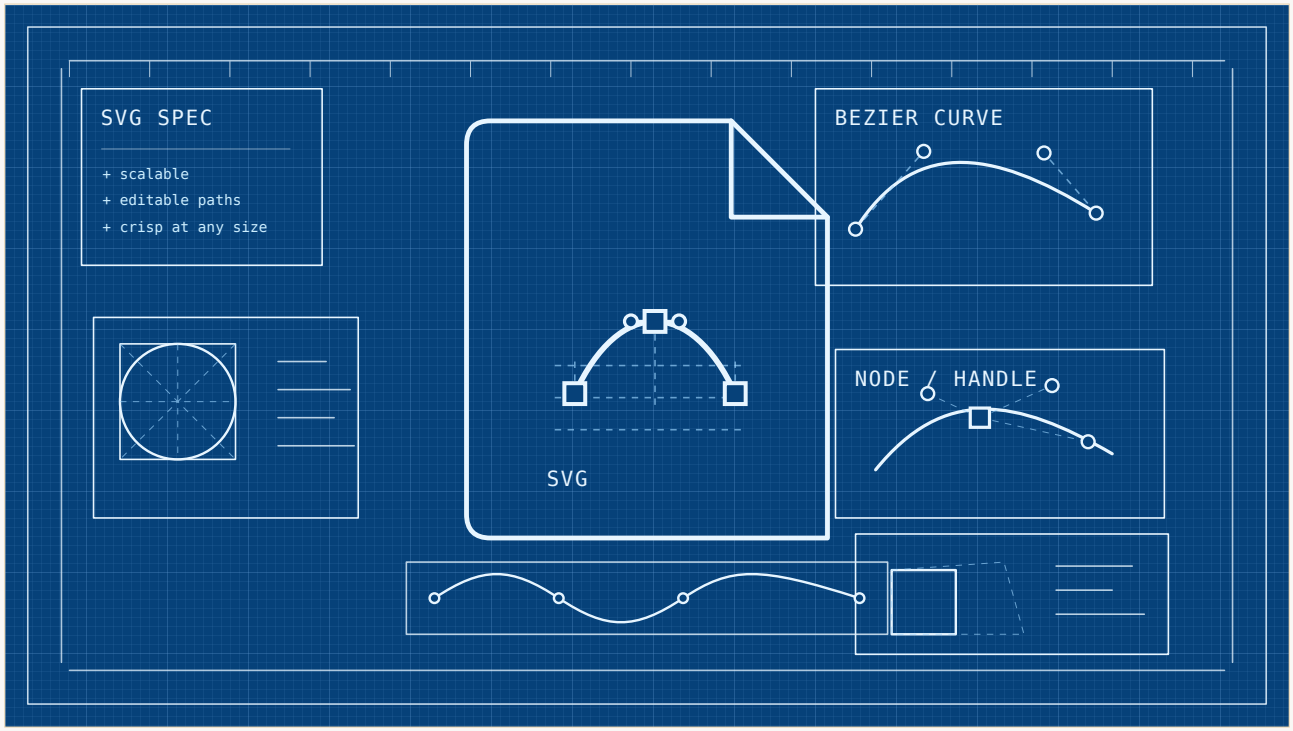
Јазичните модели што генерираат векторска графика работат на слепо — ги пишуваат инструкциите без никогаш да го видат резултатот. Нов метод му дозволува на моделот да гледа како сопственото платно се пополнува, потег по потег, а чесниот пресврт е дека само да му дадеш очи ги влошува работите: мора повторно да се обучи да ги користи. Добивката е модел што ги достигнува или малку ги надминува ривалите обучени со до дваесет пати повеќе податоци — реален, внимателен резултат на еден benchmark, не револуција.

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)

Цртање со затворени очи

Побарајте од некој од денешните AI модели да ви нацрта слика и, под површината, се случува една од две многу различни работи. Познатите генератори на слики — оние што од една реченица создаваат фотографија — директно ги „бојат“ пикселите и можат да го гледаат платното додека работат. Но постои и втор, потивок начин на цртање, каде моделот воопшто не слика. Тој пишува инструкции: *нацртај круг тука, линија таму, пополни ја оваа форма со сино*. Тие инструкции се код — истиот Scalable Vector Graphics, или SVG, што стои зад повеќето икони и логоа на веб — и предноста е реална: резултатот не е фиксна мрежа од пиксели, туку сет форми што може бескрајно да се зголемуваат, пребојуваат и уредуваат без да станат матни.



Оригинален SVG blueprint: самата слика е уредлив векторски фајл, изграден од патеки, мрежни линии, јазли и рачки наместо од фиксни пиксели.

Laura Nesso / The Clean Paper Permission on file

Проблемот е што, до неодамна, моделот што го пишува овој код цртал на слепо. Ја испуштал целата низа инструкции во еден потег — круг, линија, пополнување — без никогаш да ги рендерира за да види што направил. Замислете да скицирате лице со затворени очи: можеби приближно ќе ги ставите очите на вистинското место, но нема да забележите дека второто завршило на образот или дека формата за коса сега го покрива носот. Приближно така работеле овие модели, па често создавале код што е сосема валиден, но визуелно хаотичен.

Тим предводен од Guotao Liang сега го направи речиси комично очигледниот чекор: му дозволи на моделот да ги отвори очите. Нивниот метод, *Render-in-the-Loop*, ја рендерира

недовршената слика по секој чекор и му ја враќа на моделот пред да го нацрта следниот потег. Вистински интересниот дел не е дека ова помага — туку што морале да направат за воопшто да помогне.

> **Како се оценува цртеж?**
>
> Кога труд вели дека еден модел „победува“ друг, фер е да се праша: во што, и како е измерено? Оценувањето генерирана слика е навистина тешко — нема еден единствен точен одговор на „нацртај лаптоп“ — па полето користи неколку автоматски оценки, секоја несовршена замена за човечко гледање.
> Неколку се појавуваат и во овој труд. **FID** ја споредува целокупната статистичка „сличност“ на серија генерирани слики со реални; пониско е подобро, но ја опишува групата, не една конкретна слика. **CLIP score** прашува посебен AI дали сликата одговара на текстот на prompt-от — корисно, но ограничено од квалитетот на тој судија. **DINO**, **SSIM** и **LPIPS** споредуваат реконструирана слика со целна, од сурови пиксели до научени карактеристики.
> Ниту една од овие метрики не е „вистината“. Тие се прокси-мерки и често се поместуваат само малку. Кога ќе прочитате дека еден модел има 127.6, а друг 128.8, тоа е реална разлика во посакуваната насока — но е мал поместување, не огромна победа, и во број што само приближно се совпаѓа со она што би го кажало човечкото око. Добро е тоа да се запомни кога насловот гласи „победува модел обучен со дваесет пати повеќе податоци“.

Што направиле авторите

Проблемот што авторите сакале да го решат е самото слепо цртање. Постојните модели го третираат пишувањето SVG како чисто текстуална задача: предвиди го следниот дел од кодот од досегашниот код и никогаш не го рендерирај. Така остануваат неискористени моќните „очи“ — визуелниот encoder — што модерните мултимодални модели веќе го имаат. Render-in-the-Loop ја претвора задачата во постепен визуелен процес. По секој фрагмент код, делумниот SVG се рендерира како слика и повторно се дава на моделот, па следниот фрагмент се избира додека моделот навистина го гледа платното до тој момент.

Нивниот прв резултат е предупредување и, во нивна чест, го пријавуваат јасно: едноставно додавање на циклусот на веќе постоен готов модел не функционира. Кога на силни општи модели им давале меѓурендери без специјална обука, квалитетот не се подобрувал — туку *се влошувал* во сите тестови. Модел што никогаш не бил научен да ги користи очите за оваа задача не знае одеднаш како да го прави тоа.

Затоа најголемиот дел од работата е во обуката. Тие ги реконструираат податоците така што секој цртеж се дели на многу мали, визуелно значајни чекори — сложените форми се распарчуваат на поедноставни делови за на секоја фаза да има нешто ново

што може да се види — а потоа fine-tune-ираат отворен модел со осум милијарди параметри, базиран на Qwen3-VL, врз овие постепени секвенци. Тоа го нарекуваат Visual Self-Feedback. Додаваат и втор механизам при генерирање, Render-and-Verify: пред да се прифати секој нов потег, моделот го рендерира и проверува дали навистина ја променил сликата или само го повторил претходниот. Потезите што не додаваат ништо се отфрлаат, а кога повеќе ништо не помага, моделот добива сигнал да запре. Значајно е што сето ова работи со релативно мал сет — околу 850,000 примери, помалку од половина од податоците на еден конкурент и мал дел од податоците на друг.

Што откриле

- Обучен на овој начин, моделот црта подобро од својата „слепа“ верзија — највидливо во неуспешните случаи, каде слеп модел става око на образ или го прескокнува бараниот столбест график и наместо него црта генерички монитор.
- На стандардниот benchmark (MMSVGBench), методот е конкурентен со, а по неколку мерки и малку подобар од, силни ривали — вклучувајќи го OmniSVG, обучен со повеќе од двојно податоци, и InternSVG, обучен со приближно дваесет пати повеќе.
- Маргините се мали. Кај сетот икони, на пример, главната оценка за квалитет на слика е 127.6 наспроти 128.8 кај најдобриот ривал, а оценката за совпаѓање со prompt е 0.293 наспроти 0.291 — реална и конзистентна разлика, но тесна.
- И двата додадени елемента навистина придонесуваат: ако се исклучи специјалната обука или проверката при цртање, бројките мерливо паѓаат. Проверувањето особено спречува моделот да заглави повторувајќи го истиот потег.
- Резултатот што самите автори го нагласуваат е ефикасноста — достигнување вакви перформанси со многу помалку податоци од лидерите.

Што ова не докажува

- **Не** покажува дека само да му дозволите на моделот „да гледа“ е бесплатна победа. Напротив: сопствениот експеримент на трудот покажува дека визуелен feedback *без* повторна обука ги влошува резултатите. Добивката доаѓа од обуката, не од очите сами по себе.
- **Не** утврдува големо или одлучувачко водство. На повеќето метрики методот е многу блиску до конкурентите; „победува модел обучен со 20× повеќе податоци“ е точно, но со мали поместувања на прокси-метрики во еден benchmark.
- **Не** демонстрира општа уметничка способност. Ова е истражувачки модел со осум милијарди параметри што црта икони и едноставни илустрации на мала фиксна резолуција (224×224 пиксели), не општ дизајнер.
- Споредбата со голем општ модел (GPT-5) **не е споредба под исти услови**: тој модел не е изграден или подесен за оваа тесна задача со кодирано цртање, па победата тука кажува малку за двата модели воопшто.

- **Не** доаѓа бесплатно. Рендерирањето и повторното „гледање“ на платното на секој чекор го прават генерирањето побавно од еднократно слепо испуштање на кодот — цена што авторите ја признаваат.

Колку се убедливи доказите?

- **Цврсти** таму каде што се работи за контролирана споредба на сопствен терен. Ablation тестовите се чисти: отстранете ја обуката или отстранете ја проверката и бројките паѓаат — значи двата елемента навистина ја вршат работата што авторите им ја припишуваат.
- **Чесни за сопственото изненадување.** Резултатот дека наивниот визуелен feedback *штети* е пријавен, не сокриен, и веројатно е најинтересната поента во трудот — корисна корекција на интуицијата дека повеќе влезни информации секогаш се подобри.
- **Тенки како leaderboard тврдење.** Победите над ривали со повеќе податоци се мали и живеат на еден benchmark, изграден од авторите на еден од тие ривали. Мали маргини на прокси-метрики на еден тест-сет се сугестивни, не конечни.
- **Непроверени во голем размер и во реална употреба.** Сè тука е икони и едноставни илустрации со ниска резолуција. Дали истиот пристап ќе важи за сложен, high-resolution или реален дизајн останува идна работа — и авторите го кажуваат тоа.

Зошто е важно

Идејата во центарот на трудот е речиси засрамувачки едноставна и оди многу подалеку од цртањето. Ако програма треба да генерира нешто пишувајќи код — веб-страница, график, дијаграм, 3D сцена — може или да го напише целиот код на слепо и да се надева, или да рендерира додека работи и да ја коригира насоката. За човек тоа е природно; постојано погледнуваме во страницата. За овие модели, затворањето на циклусот е изненадувачки ново.

Она што го прави трудот вреден е ѕвездичката што ја додава. Да му дадете на моделот начин да гледа не е исто што и да го научите да гледа. „Очите“ мора да се обучат, и дури тогаш циклусот носи корист — а и тогаш добивката е реална, но мерена: постабилен цртач, не нов вид уметник. За секој што гради алатки што претвораат опис во уредлива графика — нешто што подоцна завршува во икони и илустрации на апликација — тивкиот практичен заклучок е најкорисен. Добивката тука дојде од поевтина, попаметна обука, не од повеќе податоци. Тоа е повредна лекција од уште еден поен на leaderboard.

Кратко резиме

Јазичните модели што генерираат векторска графика — уредливиот и бесконечно скалабилен код зад повеќето веб-икони и логоа — традиционално го прават тоа „на слепо“, пишувајќи ги сите инструкции без да го рендерираат резултатот. Тим предводен од Guotao Liang предлага Render-in-the-Loop: недовршениот цртеж се рендерира по секој чекор и повторно му се дава на моделот, така што следниот потег го прави додека го гледа платното. Нивниот централен и чесен резултат е дека едноставно да се направи ова со постоечки модел го прави *полош*; добивката се појавува само откако моделот е повторно обучен да го користи визуелниот feedback, со дополнителна проверка при цртање што ги отфрла потезите кои не менуваат ништо. Повторно обучениот модел со осум милијарди параметри ги достигнува или малку ги надминува конкурентите обучени со до дваесет пати повеќе податоци на стандарден benchmark — вистински резултат, но со мали маргини на прокси-метрики, за икони и едноставни илустрации со ниска резолуција. Поентата што авторите ја нагласуваат е ефикасноста: добро научено „гледање“ на сопствената работа може да надомести дел од големината на сетот со податоци.

Проверка без претерување

Што покажува трудот: Повторна обука на модел за векторска графика да ја рендерира и гледа сопствената недовршена слика чекор по чекор дава подобри и покомплетни резултати од слепо цртање — конкурентни со, а малку подобри од, ривали со многу повеќе податоци на еден стандарден benchmark.

Што е веројатно, но не е докажано: Дека „гледањето на сопствената работа“ е општо подобар рецепт од едноставно зголемување податоци; дека истата идеја ќе помогне кај сложена, high-resolution или реална графика; дека малите benchmark разлики се разлика што човек навистина би ја забележал.

Што не покажува: Дека визуелниот feedback помага сам по себе (без повторна обука штети); дека водството над конкурентите е големо или одлучувачко; општа способност за цртање; фер директна споредба со општи модели како GPT-5, кои не се изградени за оваа задача.

Главни ограничувања: Еден benchmark, делумно изграден од авторите на конкурент; мали маргини на прокси-метрики; модел од 8B, икони и едноставни илустрации на 224×224; побавно генерирање поради рендерирање на секој чекор.

Колкава доверба треба да има општ читател? Висока дека затворањето на циклусот — рендерирање и повторно гледање додека се црта — навистина помага, и дека тоа мора да се научи, не само да се вклучи. Ниска до умерена дека конкретниот модел одлучувачки ги победува конкурентите; „победува 20× повеќе податоци“ е реален, но скромен резултат од еден benchmark. Висока за идејата што вреди да се запомни: за код

што црта, гледањето додека работиш е подобро од слепо цртање.

Извори

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/