LUCID

State of AI Code Quality 2026

We ran formal verification on AI-generated code.

Average production-readiness score across AI coding platforms:

39/100

The Problem

The code compiles.
The code looks right.
The code is broken.

AI-generated code passes every traditional check:

✓ Compiles	✓ Lints clean	✓ Looks polished

But has **structural bugs** that no linter, type checker, or visual review will catch.

LUCID

Finding #1

Features that don't exist

Non-existent API endpoints

Frontend calls `/ai/analyze-scene` and `/ai/translate` — neither endpoint exists. Features silently fall back to mock data.

Analytics backed by hardcoded arrays

Dashboard renders professional charts labeled "Real-time insights" — backed entirely by static data. The useEffect just calls `setLoading(false)`.

Decorative UI buttons

5 of 6 accessibility features trigger "coming soon" alerts. Voice and camera buttons have no handlers — purely decorative.

The app appears to work. Core functionality is fake.

LUCID

Finding #2

Security that isn't there

Unprotected admin routes

All `/admin/*` routes defined with zero authentication. Any user can navigate directly to admin panels.

IDOR vulnerabilities

Any user can access any other user's data by changing the ID in the URL. Role checks exist — but ownership verification doesn't.

Auth components referenced but missing

`RoleGuard` and `PrivateRoute` used throughout routing — implementations don't exist in the codebase.

In a healthcare app, these aren't bugs. They're HIPAA violations.

Finding #3

Scaffolding that looks like features

WHAT USERS SEE

- ✓ Polished settings page
- ✓ "Real-time" analytics
- ✓ User profile management
- ✓ README: "FULLY OPERATIONAL"

WHAT'S ACTUALLY HAPPENING

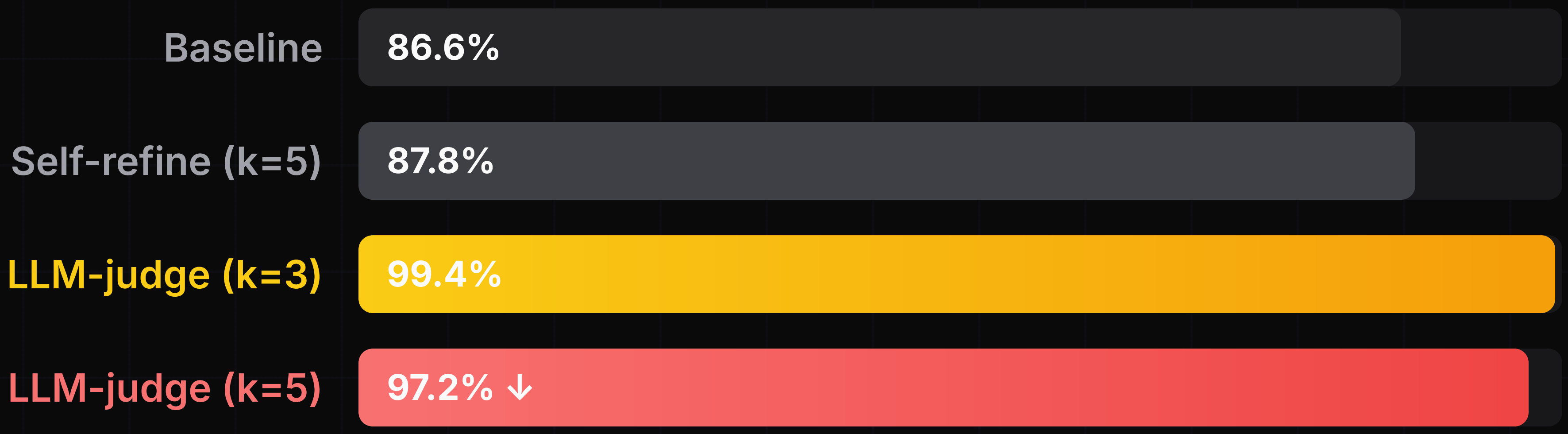
- ✕ State stored in useState only
- ✕ Data is hardcoded arrays
- ✕ Changes lost on refresh
- ✕ Name defaults to "John Doe"

The gap between "it looks right" and "it works right" is invisible to traditional tooling.

LUCID

Why This Persists

Self-refine doesn't work. LLM-judge makes it worse.



LLM-judge regresses with more iterations. False positives cause it to "fix" working code.

Self-refine plateaus at ~87%. More iterations don't help. HumanEval benchmark, Claude 3.5 Sonnet.

Formal Verification Works

LUCID converges to 100%

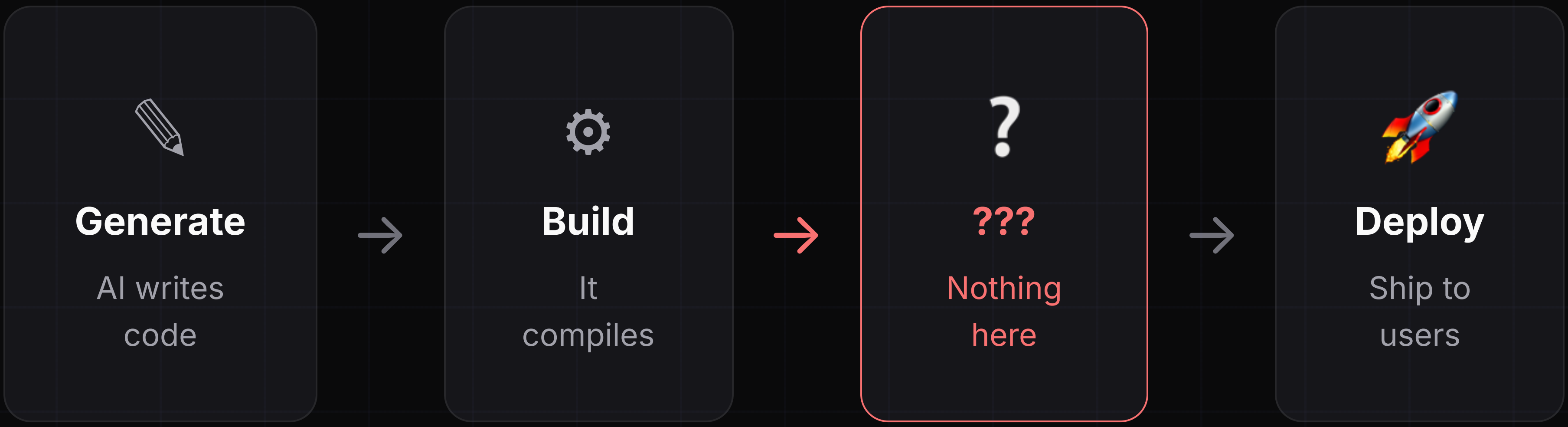
METHOD	K=1	K=3	K=5
Baseline	86.6%	—	—
Self-refine	87.2%	87.2%	87.8%
LLM-judge	98.2%	99.4%	97.2%
LUCID	98.8%	100%	100%

+36.4%
on SWE-bench
(real-world bugs)

164/164
HumanEval tasks
at k=3

The Verification Gap

There's no step between "it builds" and "it works"



That missing step is formal verification.

Linters check syntax. Type checkers check types.
Nothing checks if the code *actually does what it claims.*

LUCID

Close the verification gap.

Full benchmark report & API documentation

trylucid.dev/report

Research	Patent	Benchmarks
DOI 10.5281/zenodo.18522644	US App #63/980,048	HumanEval + SWE-bench

Ty Wells
ty@trylucid.dev