Generated: 2025-11-08 06:57 UTC

ColorLang, Explained Simply

What if code wasn't text, but color? ColorLang treats programs as pictures. Each pixel's color means an instruction. A whole image is a whole program. The computer reads the image left-to-right, decodes each pixel, and runs it.

Why do this?

Make programs easy for machines to store and move: images compress well.

Make execution visual: you can literally see your program.

Open doors to new tricks: spatial layout for parallelism, color palettes for compression.

How it works

Colors → Instructions: Hue picks the instruction family; saturation/value hold the details (operands).

Virtual Machine (VM): Reads the image, runs the instructions, and keeps track of registers and memory.

Data in Color: Some pixels are just numbers (like integers/floats) encoded in color.

Tools included

Parser: Turns images into a list of instructions.

VM: Executes those instructions.

Debugger: Lets you set breakpoints and visualize what's happening.

Examples: Small programs to test and learn from.

Compression

Because programs are images, we can compress them well:

- Palette: Reduce the number of colors.

- RLE: Compress long runs of the same color.

- Hybrid: Use both for best results.

In our demos, hybrid compression shrank files by ~99% (e.g., 168,000 bytes → ~1,000 bytes). Real results vary, but these numbers show strong potential when images have repeated regions.

ColorReact (UI on colors)

We built simple UI components (like buttons and text) that render to color grids. It's like a mini React for images. You can simulate clicks and watch the image update.

A small "mind" strip

Optionally, a 5-pixel strip encodes a simple "mood/intent" state (emotion, intent, memory recall, social cue, goal). It doesn't change program logic; it's for experiments and visualization.

What works today

Programs run. The debugger shows what's going on.

Compression works very well on demos.

UI components render and update quickly.

What's not solved yet

Pattern compression needs better serialization.

We need stronger integrity checks to verify subimages.

GPU acceleration and large-scale benchmarks are still future work.

Why this is interesting

ColorLang isn’t meant to be read by people. It’s designed for machines: compact, visual, and spatial. That’s the point. It challenges the idea that code must be text.

Where to go next

Add GPU execution for speed.

Compare to top-tier compressors on many datasets.

Build a compiler from a small text language to colors.

Improve tools that convert color programs into human-friendly summaries.