

The Soul of Neon

Teaching Robots to See Time

Cagatay Cali · March 2026 · github.com/cagataycali/neon

The problem is simple. A child watches a ball roll off a table and reaches out to catch it. She doesn't process one frozen image. She sees the *motion*—the arc, the acceleration, the moment it leaves the edge. She predicts the future from the flow of time.

Every robot today is blind to this. State-of-the-art Vision-Language-Action models look at the world through photographs. They see *where* things are, but not *where things are going*. They encode a single frame, then guess an action. It is like trying to catch that ball with your eyes closed between blinks.

Neon's insight is one sentence: *Video foundation models already understand motion—we should use them.*

Models like Qwen2.5-Omni and Cosmos-Reason2 have watched millions of hours of video. They have learned, implicitly, that cups fall when pushed off edges, that doors swing on hinges, that hands reach before they grasp. This temporal knowledge—physics, dynamics, cause and effect—is exactly what a robot needs. It is sitting there, pre-trained, waiting.

So we do something radical in its simplicity. We take a 7-billion-parameter video model. We **freeze** it entirely. On top, we train a *tiny* action decoder—just 6 million parameters, 0.08% of the total—that translates the video model's rich temporal understanding into 29 joint commands for a humanoid body, 16 timesteps into the future. The video model sees. The decoder acts.

The mathematics are beautiful in their economy. The decoder uses techniques from a competition to build the smallest possible language model: $\text{ReLU}^2(x) = \max(0, x)^2$ for smooth gradients, RMSNorm for cheap stability, learnable residual scales $h' = f(h) + \alpha \cdot h$ that let the network decide how much to trust its own frozen backbone, and soft-capping $c \cdot \tanh(x/c)$ that never kills a gradient at the boundary. Every trick matters when your entire trainable model fits in 25 megabytes.

The data story is equally radical. A robot built for a specific body usually needs data *from that specific body*. We break this constraint. Using relative actions—displacements computed in the gripper's own coordinate frame—the same reaching motion produces the same numbers whether performed by a Franka arm, a tabletop SO-100, or our Unitree G1 humanoid. This is provably invariant to workspace origin and robot geometry. Suddenly, every robot dataset in the world becomes training data for our humanoid.

We call this *data soup*: seven different source types—LeRobot demonstrations, Agibot bimanual manipulation, Cosmos synthetic generation, stereo depth from kitchen cameras, 50,000 spoken voice commands, and G1 teleoperation—all mixed by weighted sampling into a single training stream. The model learns from all of them simultaneously.

And then the robot speaks. Neon doesn't just hear voice commands through its Whisper audio encoder—it speaks back through PersonaPlex, narrating its intentions: *"I see the red cup. Reaching for it now."* A humanoid that explains what it's doing, in real time, as it does it.

The full system—video backbone, action decoder, audio in, speech out—runs at 50 milliseconds per step on a Jetson Orin the size of a credit card. The model, training code, data pipeline, 80 tests, a full paper, and this page are all released under the MIT license.

This is what we believe: The path to capable robots does not require training ever-larger models from scratch on expensive robot data. It requires *connecting* the temporal understanding that already exists in video models to the physical world through minimal, elegant interfaces. The video model is the brain. The action decoder is the spinal cord. The data soup is the experience of a thousand different bodies, unified.

Neon is 18 Python files, one idea, and an invitation: teach your robot to see time.

`pip install neon-vla` | MIT License | 80 tests passing | <https://cagataycali.github.io/neon>

"The difference between seeing a photograph and watching a video is the difference between knowing and understanding."