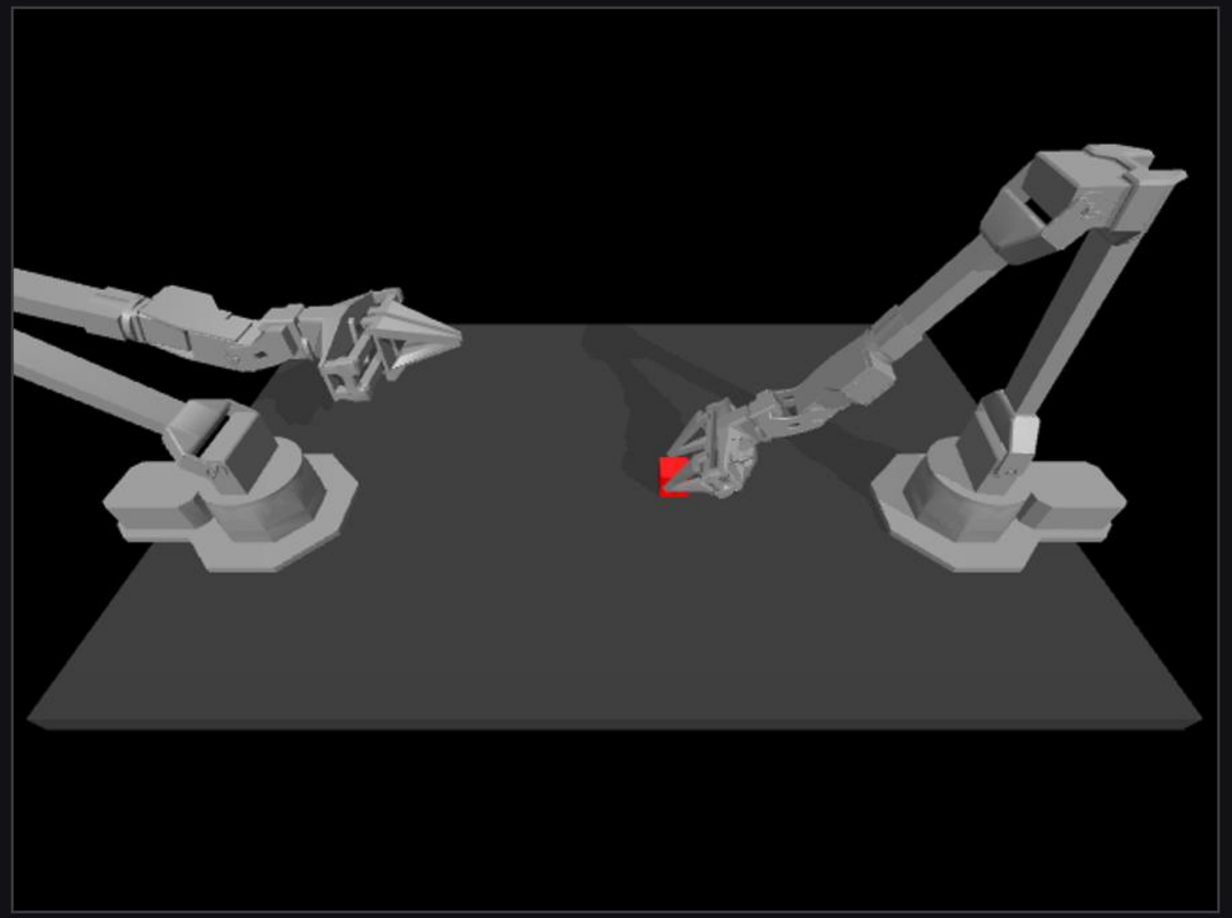
Bimanual VLA on Intel

AI Infra Summit Hackathon

Intel Bimanual VLA Manipulation · Speechmatics bonus track

A measurement, not a demo.



github.com/peacestate/aiinfra-vla

THE PROBLEM

A voice-controlled bimanual arm needs two things, not one

1. It has to actually do the task.

Bimanual manipulation: one arm picks up an object, hands it to the other arm.

2. Its behavior has to actually depend on the words.

Otherwise the voice interface is decoration — the robot would do the same thing no matter what you say.

Most projects measure #1 and assume #2.

This project measures both — rigorously — and reports what it actually finds.

APPROACH

LangACT: language, added to ACT for free

ACT already builds an internal feature from 4 sources:

[latent, robot state, env_state (UNUSED), image features]

We write a sentence embedding into that unused slot.

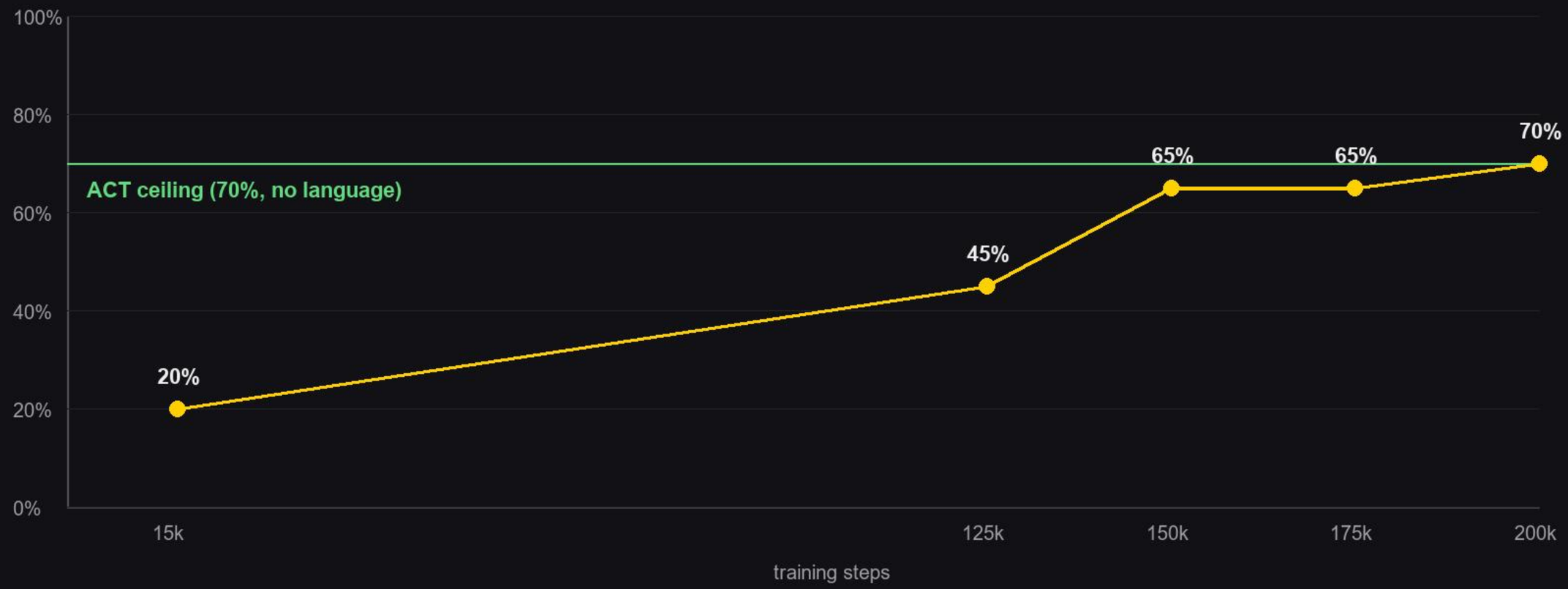
Result:

- 0.2% more parameters (51.6M → 51.8M)
- 0 ms inference cost — instructions are known strings, embedded once and cached, never re-run
- No custom architecture — stock, unmodified ACT

Trained 15,000 → 200,000 steps across 5 chained Kaggle T4 sessions (each resuming the last checkpoint).

RESULTS

Task performance: 20% → 70% over 200k steps



THE MEASUREMENT THAT MATTERS

Does language actually steer behavior?

The instruction-swap test: run the SAME task, feed the WRONG instruction. If language matters, success should collapse.

checkpoint	normal instruction	swapped instruction	verdict
125k steps	9/20 (45%)	12/20 (60%)	WRONG DIRECTION
150k steps	13/20 (65%)	13/20 (65%)	IDENTICAL SEEDS
200k steps	14/20 (70%)	14/20 (70%)	IDENTICAL SEEDS

At every checkpoint tested, changing the sentence never changed a single episode's outcome — even at 200k, where the model ties ACT.

ROOT CAUSE

Why? A dataset confound, not a training problem

In every training episode, the task and the instruction are
100% correlated with the visual scene:

- Cube-transfer episodes are ALWAYS paired with the cube instruction
- Insertion episodes are ALWAYS paired with the insertion instruction

The model can solve the task perfectly from vision alone.

Nothing in training ever rewards reading the sentence.

Confirmed at the numeric level too: exporting to OpenVINO,
swapping the instruction DOES move the raw output (2.5-3.0e-2)
— the weight is real and reacts. It is just too small to survive
100+ steps of closed-loop feedback and ever change the outcome.

INTEL OPTIMIZATION

OpenVINO + INT8: official benchmark_app numbers

runtime	median	throughput
fp32, CPU	239.1 ms	3.92 FPS
INT8, CPU	195.9 ms	4.20 FPS
INT8, Intel iGPU	**71.0 ms**	**13.91 FPS**

INT8 weights: 132.6 MB → 33.7 MB (3.9x smaller)

Reproducible with: benchmark_app -m ir/langact_int8.xml -hint latency -d GPU

Hardware disclosure: 12th Gen Core i3 (Alder Lake), CPU + Intel UHD iGPU only — no NPU.

SPEECHMATICS — VOICE CONTROL

Since in-model conditioning is inert, we route instead

Speech → Speechmatics STT → keyword-matched intent → runs the matching trained skill → Speechmatics TTS confirms the true outcome

"pick up the cube and pass it to your other hand"

cues → {transfer: 3, insert: 0} → MAP [transfer]

RUN → SUCCESS (241 steps) → "Task complete. Cube transferred."

"insert the peg into the socket"

cues → {transfer: 0, insert: 4} → MAP [insert]

SKILL NOT READY (measured 0/20 on its own task) — declines honestly

The spoken words genuinely determine what the arm does — or honestly doesn't — through explicit, auditable routing.

CONCLUSION

What this project actually shows

- **Bimanual manipulation works**
  70% success, ties ACT, on real closed-loop rollouts
- **Language conditioning is free but inert on this dataset**
  diagnosed as a dataset confound, not a training-length problem
- **Voice control genuinely works**
  via explicit routing, not by pretending the model reasons over language
- **Real Intel hardware, real speedup**
  541ms → 90.5ms (6x), no NPU — stated plainly

"The pitch is a measurement, not a demo."

github.com/peacestate/aiinfra-vla