01

YOLO AND VISION-LANGUAGE-ACTION POLICIES FOR ROBOTICS ARM

Final Projects

Pree Simphliphan

prees26@bu.edu

Lion (Xingjian) Jiang

starlion@bu.edu

ENG EC 535: Introduction to Embedded Systems
Boston University College of Engineering

MOTIVATION BEHIND USING LOW-COST ROBOTIC SOLUTIONS

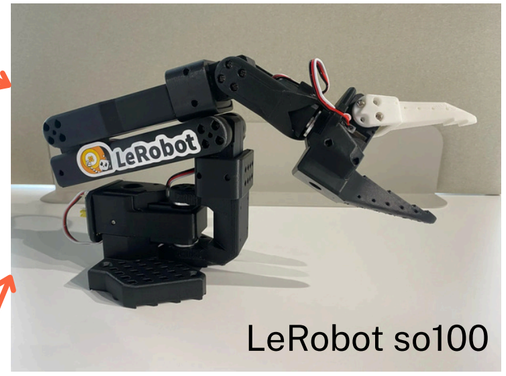
Augmenting Low-Cost Robotic Arms with YOLO-Based Perception and Vision-Language-Action Policies

Workspace clutter and messy

Prof Introduced in class

3D printed

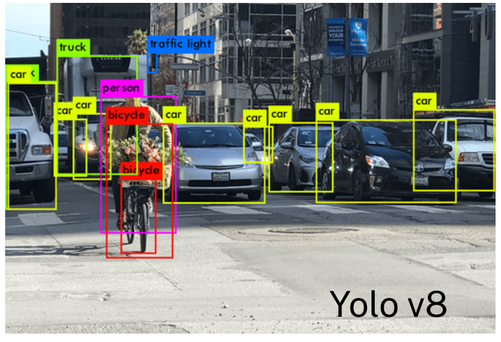
assembly easy



LeRobot so100

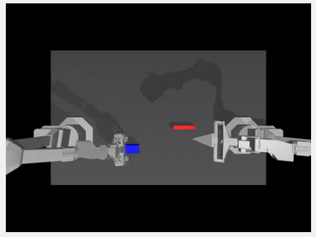
Costly Industrial robotic solutions

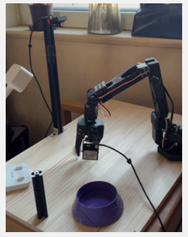
+



Yolo v8

Improve Performance on more generalized cases





- ACT vs Diffusion Policy
- Generalization Capability Limited

Goal

- Achieve generalization without retraining
- Integrate YOLOv8 for object detection and vision-language-action diffusion policies.
- Develop a low-cost, adaptable robotic arm solution.

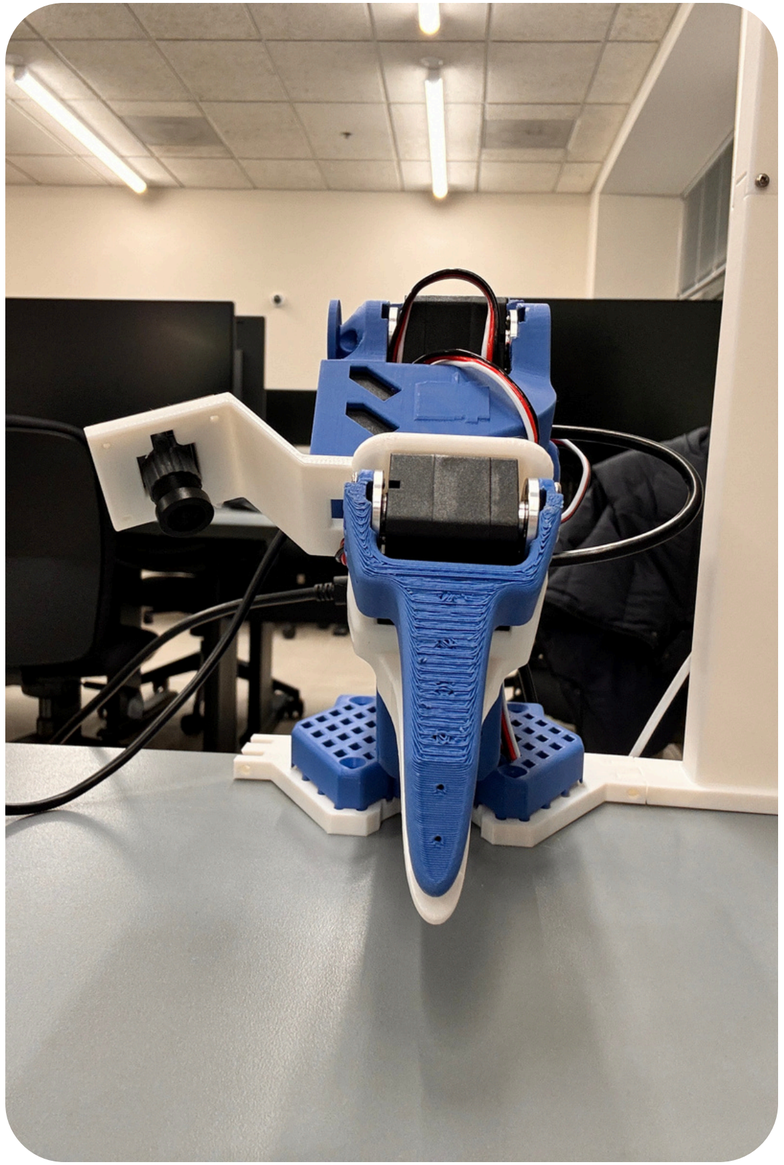
03

Methodology

03

Main Parts

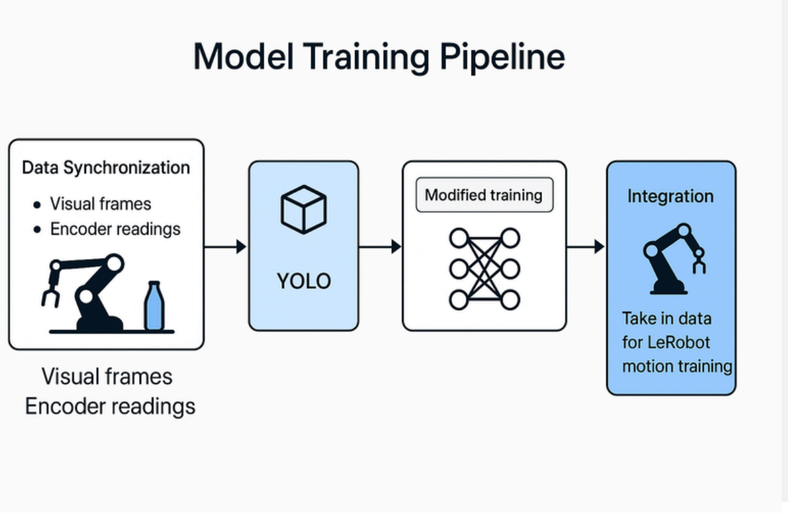
- ESP32-S3 with a camera detects objects, send each frame through web server to PC
- Using YOLO model to computes their center coordinates, and a LeRobot framework drives the robot arm

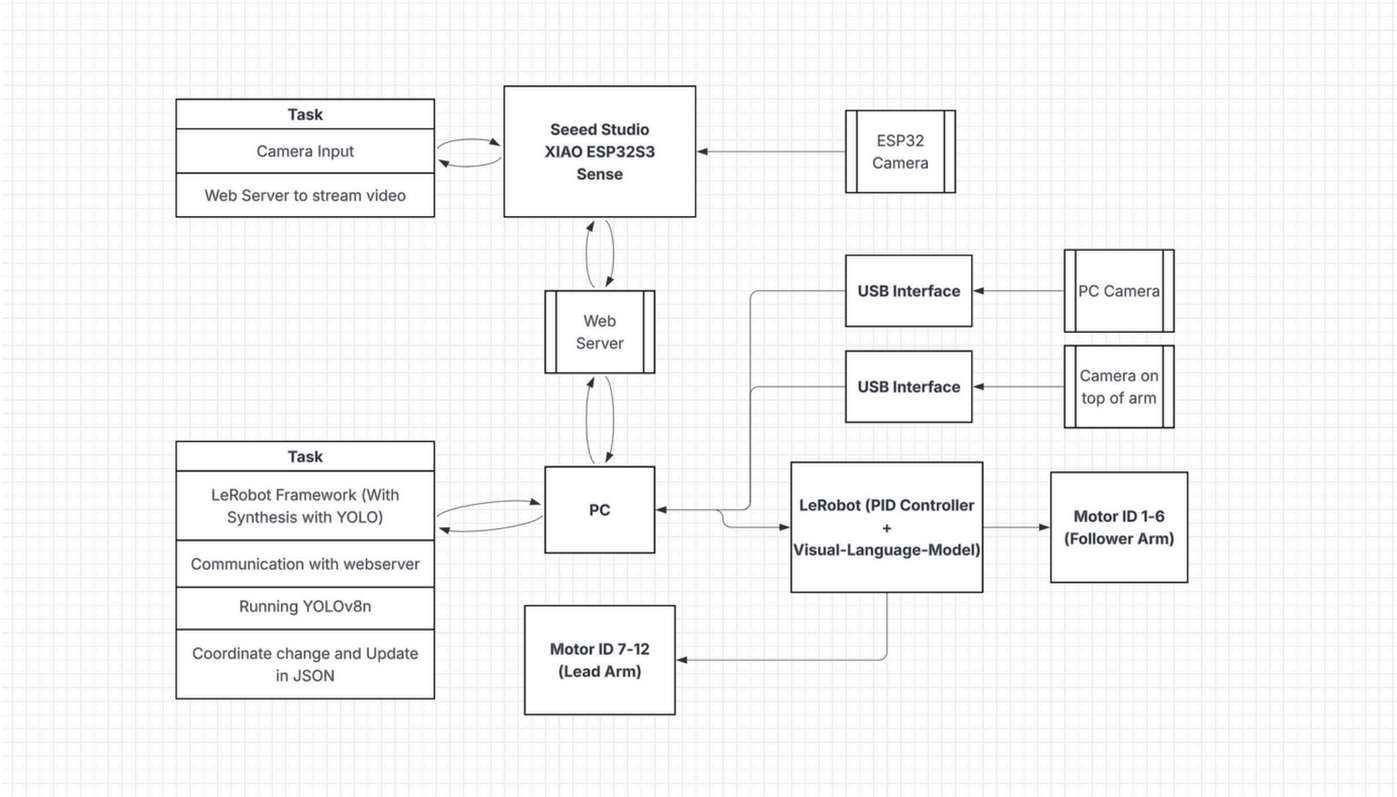


Back to First Page

05

Our System Diagram





Technical Challenges and Solutions

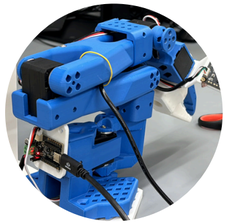
- ESP32 computational limitations → Stream video data to PC.
- Integrate YOLOv8 for object detection and vision-language-action diffusion policies.
- Servo motor control without force feedback → Software and visual training.

06

Hardware

Seeed Studio XIAO ESP32 S3 Sense

2.4GHz Wi-Fi, BLE 5.0, OV2640 Camera Sensor, Digital Microphone, 8MB PSRAM, 8MB Flash



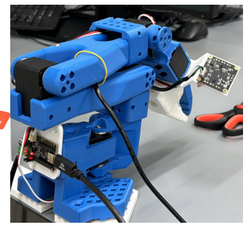
6-DOF SO-100 Robotics Arm x2

STS3215 Servo Motors: 30kgcm
compatable with LeRobot Framework
2 camera modules

Training Setup

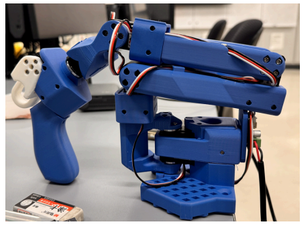
Intel U7 265k + RTX4070
Training the ACT and Diffusion Policy with our collect Dataset

+



Follower Arm
(main actuator)

Bird View esp32cam
for YOLO



Leader Arm

||

Full View

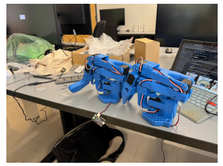
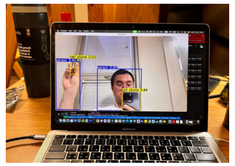
Software

YOLOv8n + Ultralytics + OpenCV	ESP-IDF	LeRobot + Python 3.10 + Communication
• YOLOv8n: bottle, scissors • Ultralytics: Load YOLO • OpenCV: handle frame from camera	• ESP-IDF v5.4, integrating esp_camera • Captured frames from esp_camera_fb_get() • WiFi Configuration • FreeRTOS for scheduling	• Training and analyzing result on LeRobot website • HTTP Communication • Serial Communication (UART)

Deployment

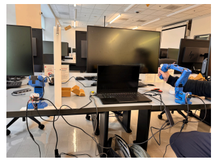
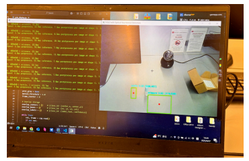

Phase 1

Configuring the LeRobot and ESP32-S3 with camera modules and web server

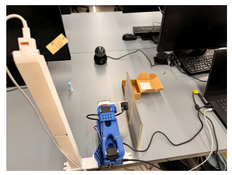
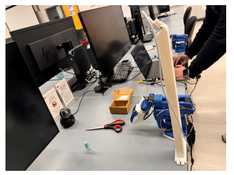
Phase 2

Finish assembly and Fine-tuning YOLOv8n model and compare with original model

Phase 3

Integrating LeRobot with YOLO data and train the dataset with teleoperation

Phase 4

After training model, testing the result of the model with our setup with one object pick-up

09

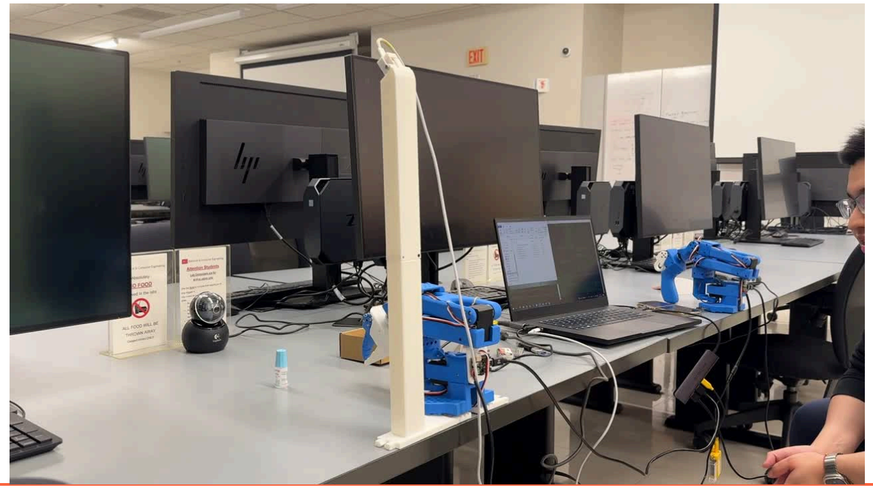
Results

Success Rate

Back to First Page

1 out of 2

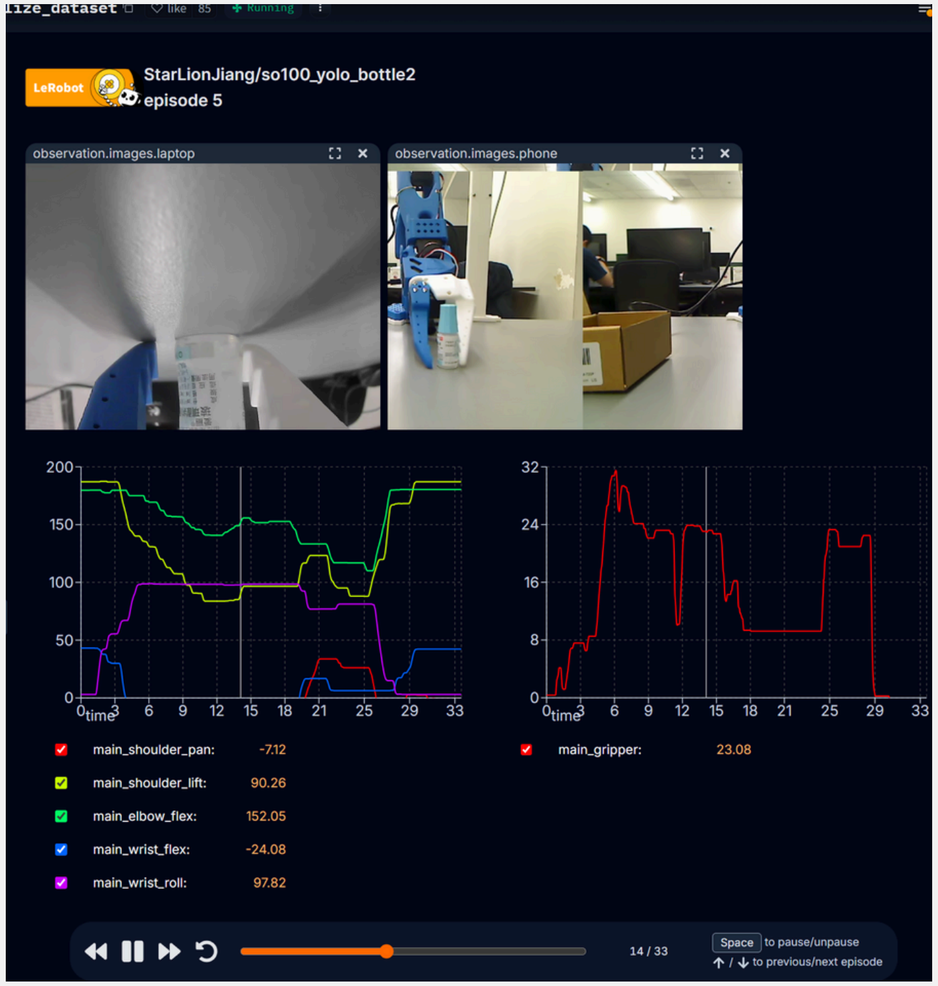
50.00%



Results

ACT vs Diffusion Policy

- 50 teleoperations



Back to First Page

12

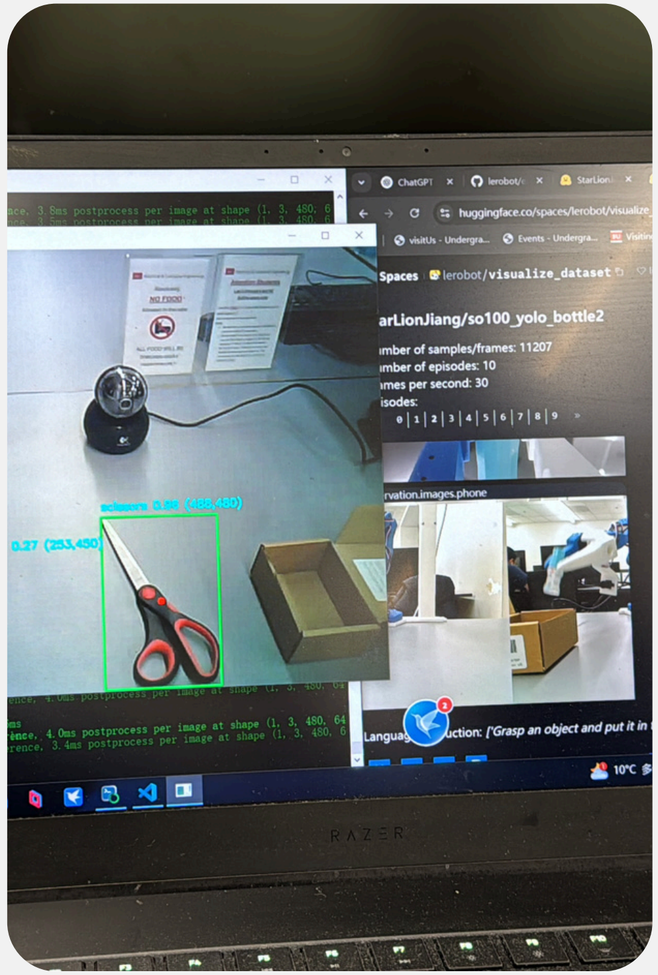
Conclusion and Future Plans

Conclusion

Achievement: Integration of modern vision models and data-driven learning to extended the capability of a basic robotic arm.

Skills: Assembly, 3D printing, Firmware Development

Concepts: Memory, Computer Vision, Communication in Embedded Systems, Scheduling, Low Power Management



Future Plan

- Fine-tuning a YOLO model specific to environment for even higher accuracy.
- Transfer result from YOLO running in ESP32 (relying on esp32 only)
- Integrating table organizer that is ready to go commercial with any general setup without predefined training

Back to First Page

Thank you!