DESIGN AND CONTROL OF A FETCHING QUADRUPED

Individual Progress of Teng Hou

By

Teng Hou

Individual Progress for ECE 445, Senior Design, Fall 2025

TA: Xuekun Zhang

April 17, 2025

Project No. 3

Abstract

In this phase, I led the development and deployment of the vision perception and target-tracking subsystem. I began by assembling and annotating a specialized image dataset for the fetching task, applying data-augmentation techniques to bolster robustness across diverse lighting and environmental conditions. Building on a pretrained YOLOv8n backbone, I designed an end-to-end training pipeline—conducting hyperparameter optimization, tailoring the loss function, and employing multi-scale training—which yielded a mAP50 $\geqslant$ 95% on the validation set. To satisfy the real-time requirements of the quadruped platform, I applied model pruning and quantization, then deployed the optimized network onto an embedded computing unit. Leveraging OpenCV and ROS interfaces, the system sustains online inference at over 20 Hz. Finally, I integrated the vision module with the robotic-arm control stack and validated detection accuracy and pose-estimation stability in live object-retrieval trials, laying the groundwork for the autonomous fetching capability.

Contents

1. Introduction

1.1 Team Project Overview

Our project's goal is to build an object fetching system that combined with a robot dog and our self-built robot arm. The function pipeline that we want to implement is recognizing the object, tracking the object, sending position and depth message from the dog to arm and fetching the object. Our approach integrates a manipulator with the quadruped using external sensing feedback, allowing the robotic system to autonomously retrieve objects. The robotic arm is designed to be compact and lightweight, ensuring minimal impact on the dog's mobility while enhancing its capability.

1.2 Individual Responsibilities and Role in the Greater Project

As the team's ECE specialist, I am responsible for the vision-recognition subsystem that enables our quadruped to autonomously detect and localize objects for fetching. This entails curating and annotating a diverse image dataset, selecting and fine-tuning a deep-learning model (YOLOv8n) [1] through hyperparameter optimization and multi-scale training to achieve mAP50 $\geqslant$ 95%, and applying pruning and quantization to ensure real-time inference ($\geqslant$ 20 Hz) on the robot dog's embedded computing unit. I then integrate the optimized detection pipeline with the robot's camera and control stack via ROS and OpenCV, collaborating with the mechanical-design and motion-control teams to validate detection accuracy, pose estimation stability, and seamless handoff to the manipulator module.

2 Individual Design Work

2.1 Design Consideration

2.1.1 Vision Sensor Selection

Although adding an external, high-resolution camera promised wider field-of-view and improved image fidelity, the trade-offs in mass, power consumption, and mechanical mounting complexity proved prohibitive. Mounting a second camera would have increased payload weight by over 200 g, reducing the quadruped's battery life by an estimated 10 %, and necessitated a custom gimbal and ROS‑level time-synchronization across two image streams. By contrast, the dog's onboard RGB-D sensor already provides a synchronized depth and color feed at 30 Hz with minimal latency, seamlessly integrates with its embedded computing unit, and avoids additional calibration steps. Leveraging the native camera thus simplified both hardware and software integration while preserving sufficient resolution (640×480) and frame rate for robust detection and tracking in our indoor test scenarios.

2.1.2 Target Object Selection

Our initial prototype targeted small spherical objects (e.g., foam balls) to demonstrate the system's fetching capability; however, the ball's uniform curvature produced ambiguous key points and hindered reliable pose estimation, causing the gripper to slip or misalign. Furthermore, the reflective surface of standard sports balls introduced specular highlights that confused the detector under variable lighting. To address these issues, we switched to using a smartphone as the retrieval target. The phone's flat, rectangular geometry yields well-defined edges and corners for more accurate bounding-box regression and 6-DoF pose estimation, while its matte screen minimizes glare. This choice not only improved grasp stability but also better reflects practical object-retrieval tasks in consumer environments.

2.2 Diagrams

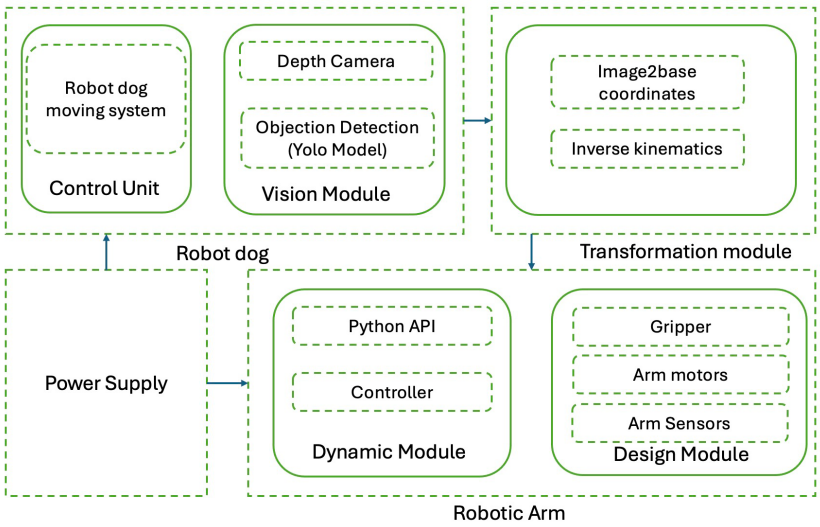

Figure 1. The whole block diagram of the project

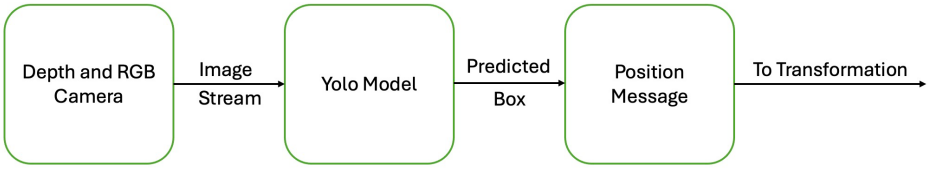

Figure 2. Detailed vision module diagram

2.3 Testing Verification

For Testing whether my part could act as I expected, I design three main testing and verification method.

2.3.1 Yolo Model

For this part, I just want to verify that my yolo model is well-trained and could detect and recognize the object. To achieve this, after training the model, I find around 1,000 new images with mobile phone to test whether my model could detect them. And the result showed that even under ambiguous environment, my model could also recognize the phone.

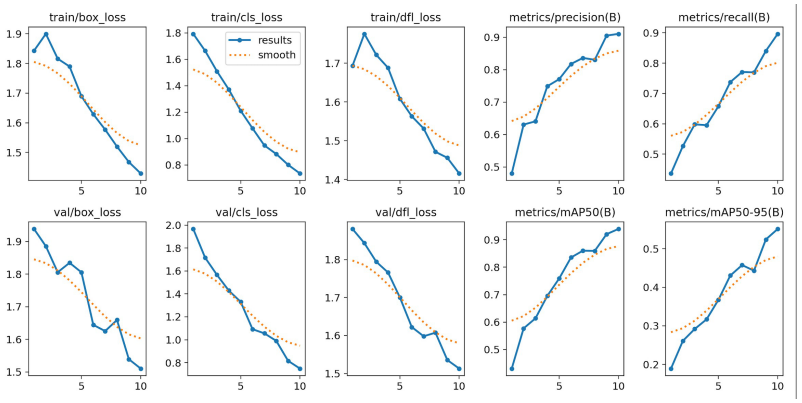

Figure 3. Evaluation metrics of Yolo model (I set x-axis to be number of training round divided by 10)

Besides the metrics, I also test the model behavior on the testing dataset.

Figure 4. Testing result on test dataset

2.3.2 Object Detection and Tracking

To validate this functionality, I have implemented a pipeline that captures the live camera feed from the robot dog's onboard sensor in real time, automatically slices each full-resolution frame into a grid of smaller tiles and then queues those tiles for object detection. I've already confirmed end-to-end performance on my local workstation—using a USB webcam as a stand-in for the quadruped's camera—by pulling the MJPEG stream via OpenCV's [3] Video Capture interface, splitting every frame into uniform patches, and dispatching them to the detection routine. The core Python script (currently under active development) handles stream acquisition, frame buffering, tile generation, and synchronization with the inference engine; I'm now refining its ROS topic subscription logic, error-handling, and tiling parameters to ensure robust, low-latency operation on the embedded computing unit.

2.3.3 Message Transformation

For this part, I need to verify that I could transform the predicted box message to the massage that could be accepted by the robot arm (which means the angle and depth). Actually, this transformation part is responsible by my teammate, so I just need to generate the predicted box message and store it on the device. This part could be easily achieved by using the parameter of Yolo. After one prediction, I could have txt file which contains predicted object box position.

3. Conclusion

3.1 Self-assessment

To date, my individual contribution—spanning dataset curation, annotation, model selection, hyperparameter tuning, pruning, quantization, and preliminary integration—accounts for approximately 25 % of the project's total workload. I successfully trained a YOLOv8n-based detector to mAP50 $\geqslant$ 95 % and implemented a real-time tiling pipeline two days ahead of the original schedule, creating valuable slack for cross-module testing. Throughout development, I maintained close coordination with mechanical and control teams to address integration challenges, such as latency spikes during streaming and synchronization discrepancies between vision and actuation loops. While initial model convergence required three full training iterations, subsequent pipeline optimizations proceeded smoothly, and I have consistently met or exceeded weekly sprint goals. This proactive pacing sets a strong foundation for end-to-end system validation.

3.2 Plans for Remaining Work

By May 1, I will complete the full verification of the Object Detection and Tracking pipeline, ensuring reliable real-time performance and accuracy across diverse scenarios. Upon finishing this verification, I will pivot to support my teammate on the coordinate-transformation and data-migration tasks—defining and implementing the necessary frame conversions, message mappings, and ROS bridges to seamlessly hand off perception outputs to the manipulator. I will then work closely with the control team to refine the perception-to-actuation interface, validating that transformed pose data drives precise, robust arm motions. This collaborative effort will elevate the system's integration fidelity and bring us one step closer to our autonomous fetching demonstration. Finally, this plan encompasses the subsequent mock demonstration, during which I will collaborate with my teammates to produce a fully integrated physical prototype capable of autonomously fetching objects.

4. Ethics and Safety

We always adhere to what we have committed to in our proposal. Ethics:

We are committed to upholding the highest ethical standards and ensuring the integrity of our project by strictly adhering to the IEEE Code of Ethics [2]. In doing so, we clearly specify the intended application and operational constraints of our system (IEEE Code 2) and accurately describe workspace and payload capabilities (IEEE Code 5). We also recognize and promptly address any technical deficiencies, while properly crediting all team members for their contributions (IEEE Code 6). To protect personal privacy, our vision system avoids collecting unnecessary information (IEEE Code 1), and we consider the environmental footprint of our design and production processes—particularly in relation to 3D printing (IEEE Code 1). Furthermore, we establish open communication channels for reporting concerns and issues (IEEE Code 7). Throughout our work, we prioritize public safety and well-being, remain vigilant about potential negative social or environmental impacts, and maintain honesty and integrity in all professional activities, thus avoiding unethical conduct such as bribery or illegal actions.

Safety:

To ensure the safety of both team members and equipment, we implement high-voltage protection when working with motor drivers and power systems and conduct regular checks of all power cables and connections. We provide clear instructions for operating the robot dog and manipulator arm, along with an emergency stop mechanism that can immediately halt operation if necessary. This is further supported by torque limiters and mechanical stops to prevent overload and over-extension. Regular maintenance of all mechanical components and a limit on the end effector's maximum grip force ensure safe handling of objects. Dedicated testing areas in the lab, along with adequate ventilation—particularly during 3D printing—help maintain a secure working environment. We also prohibit team members from working alone in the laboratory, enforce strict guidelines for handling and charging batteries or other potentially hazardous materials, and document these measures in standard operating procedures to ensure consistent safety protocols throughout the project.

References

[1] Ultralytics. (2025, March 16). Home. Ultralytics YOLO Docs. https://docs.ultralytics.com/

[2] IEEE, IEEE Code of Ethics, Online, Available: https://www.ieee.org/about/corporate/ governance/p7-8.html, 2020.

[3] Vishal, R & Balan, Siva. (2023). Motion Capture and Frame Extraction from Video Using OpenCV. 10.3233/ATDE221309.