A DODGEBOT SYSTEM

By

Chiming Ni

Feiyang Wu

Kai Wang

Nichen Tian

Yiyang Bao

Final Report for ECE 445, Senior Design, Spring 2025

TA: Lijin Ma

16, May 2025

Project No. 27

Abstract

DodgeBot is an autonomous robotic training system designed to enable solo dodgeball practice. By combining a friction-wheel launching mechanism, a dial-plate ammunition loader, and advanced computer vision with You Only Look Once (YOLO) algorithms, the system provides interactive and realistic dodgeball training. It accurately launches dodgeballs at consistent speeds of approximately 2 m/s, dynamically adjusting trajectories based on player movements tracked with a high accuracy. This project successfully demonstrates the potential of robotic systems to enhance athletic training experiences through precise targeting, interactive gameplay, and real-time adaptive responses.

Contents

1 Introduction

1.1 Background

Dodgeball is one of the types of very exciting sports where agility is essential and quick aiming for target is required. Conventional training methods of dodgeball rely heavily on human partners for both throwing and dodging and for an individual athlete to efficiently and effectively train alone is a significant challenge.

1.2 Problem Statement

Current automated methods, such as tennis ball projectiles or launching devices, are limited in their versatility, accuracy, and interactivity, three features that are necessary for simulating authentic dodgeball experiences. This gap limits athletes' successful self-training, highlighting the need for a smart, interactive and adaptive training constellations.

1.3 Proposed Solution

To the best of our knowledge, we present the first rhythmic solo training exercise embodied by an autonomous intelligent robot that has been developed for dodgeball practice: DodgeBot. DodgeBot implements a pair of motored friction wheel for the launcher and a turret mounting the dial-plate for ammunition loading, which builds a highly accurate and reliable ball launching system that can potentially keep the ball speed at a steady speed of approximately 2 m/s, while the system is controlled by the state-of-the-art computer vision and deep learning algorithm of YOLO for real-time player figuring out with a higher than 80% precision and trajectory planning and ball shooting dynamically according to the rapid adaption of player's movement.

1.4 Key Features

Key features of DodgeBot are accurate ball launcher, dynamic trajectory adaptation to player movement, and computer vision-based tracking. These aspects make an engaging training environment that mirrors the reality of dodgeball play.

1.5 Report Overview

In this paper, we describe the detailed design and validation of the DodgeBot system. The second chapter includes the design flow, the components selection, and system integration. This chapter presents design validation where we demonstrate that DodgeBot satisfies the established performance requirements via exhaustive testing. A breakdown of costs and labor is provided in Chapter 4. The final chapter discusses our contributions, limitation/confidence of development, ethical issues and future work.

The DodgeBot system greatly augmented the individual training of dodgeball athletes, which provides an interactive, adaptable, and realistic training environment. DodgeBot laps the competition by combining gym-quality ball launching with cutting-edge tracking capabilities; this is the new standard for self-play sports training technology.

2 Design

2.1 Subsystem Overview

2.1.1 Mechanical Subsystem

The design prioritizes rapid response and high-precision launching capabilities under dynamic conditions. To achieve this, the robot adopts a modular framework consisting of a dual-axis gimbal, a friction-wheel-based launching mechanism, a helical-feeding ammunition system, and an integrated sensory array.

The yaw mechanism consists of three horizontally arranged plates supported by brass spacers, with the yaw motor directly coupled to the top plate for continuous rotation. A hard stop prevents over-rotation when cable length is insufficient.

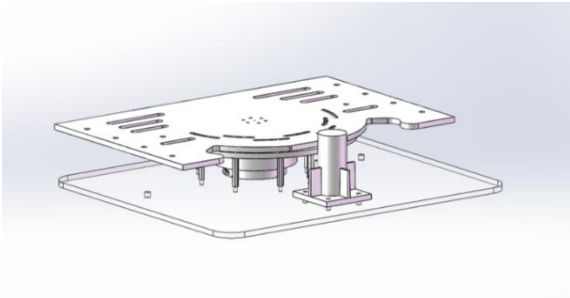

Figure 1. yaw axis

The pitch mechanism features a clamping plate structure, with a connecting rod linking the motor to the gimbal. Thrust bearings are deployed to reduce resistance at the pivot joints, enabling precise ±30° elevation control.

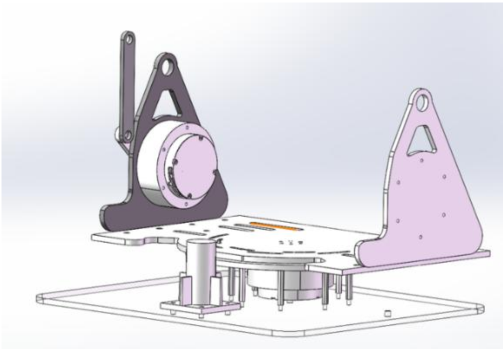

Figure 2. pitch axis structure

The feeder consists of a rotating disc housed within a PLA container, capable of storing up to 8 Wiffle balls. A centrally mounted M3508 motor drives a custom-designed helical pusher, which guides balls along a spiral groove toward the launcher inlet.

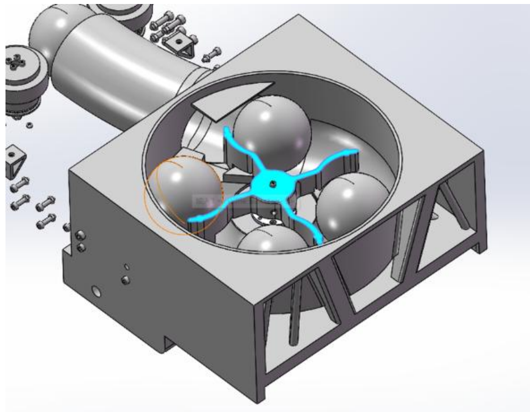

Figure 3. ball feeder arm, a pusher

The launcher features dual M3508 motors spinning rubberized friction wheels at calibrated distances (68.216 mm) to match the 72 mm diameter of Wiffle balls, by squeezing the balls into the friction wheel launcher.

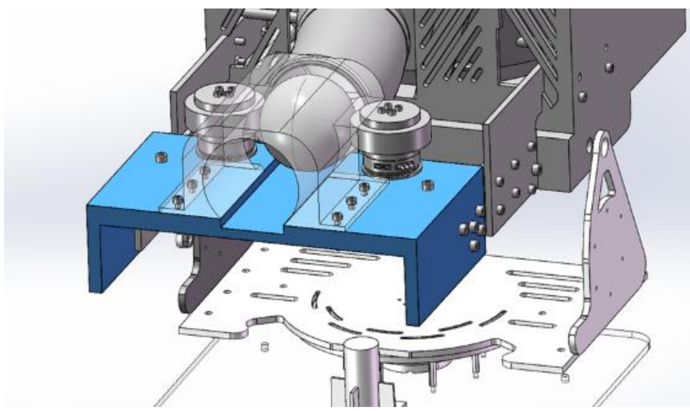

Figure 4. motor holder and the barrel structure

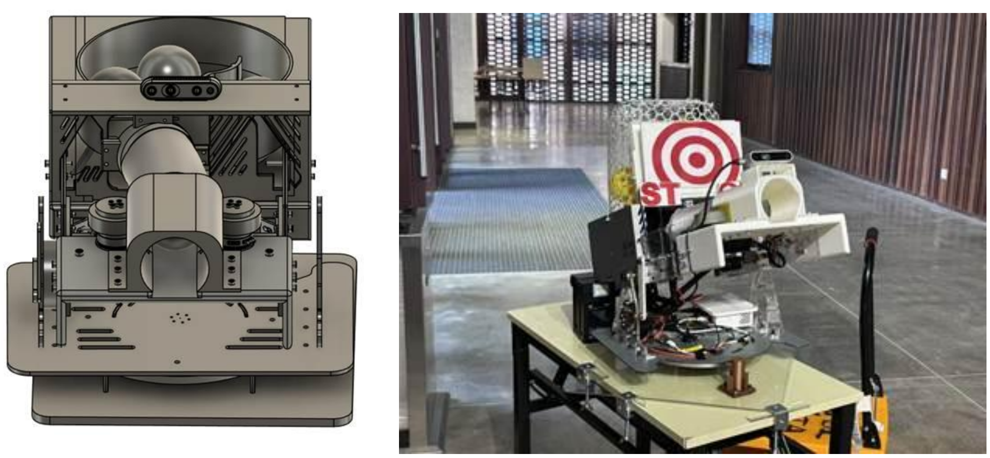

Figure 5. The final version of Dodgebot

2.1.2 The DodgeBot Control Subsystem

The DodgeBot Control Software is built upon ROS2 and ROS2-Control software stack.

ROS2 (Robot Operating System 2) [1] is an open-source robotics middleware that provides tools, libraries, and frameworks for building complex, real-world robotic systems. ROS2 uses Data Distribution Service (DDS) as its default middleware for reliable, low-latency, and scalable messaging. ROS2 Control framework is built on top of ROS2 but provides real-time control and optimization.

The overall software design of our robot includes four layers, perception layer, decision layer, execution layer, and decomposition layer. Perception layer is related to robot perception, including robot vision related camera SDK and the board IO code to detect dodgeball hit. Decision layer includes all of our YOLO vision inference, human tracking, and gaming logic functionality. Execution layer contains all ros2-control related real-time hardware including CAN motors, while decomposition layer contains non ros2-control components that do not require rigorous real-time performance including remote control.

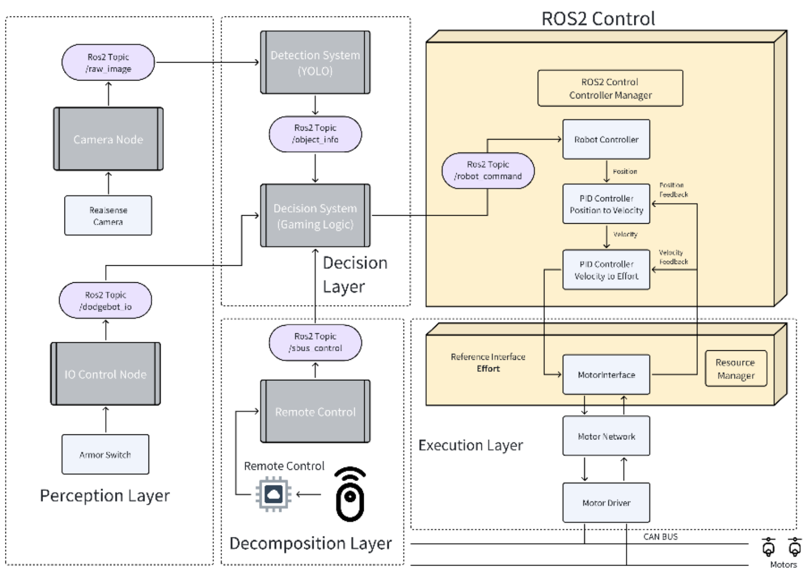

Figure 6. Block Diagram of Software Architecture

The Realsense Camera in the perception layer serves as the image provider for the YOLO model and it consumes the images. The tracking part in the decision layer outputs desired yaw and pitch angles and send them to the game logic node, while the remote control is also sending out the same command. Game logic will decide which to use and send the command to the execution layer, after which the motors will be at the desired position.

The gaming logic, or how DodgeBot interacts with players, is encapsulated in the decision system. The flow-map for the gaming logic of DodgeBot is illustrated in Figure 7.

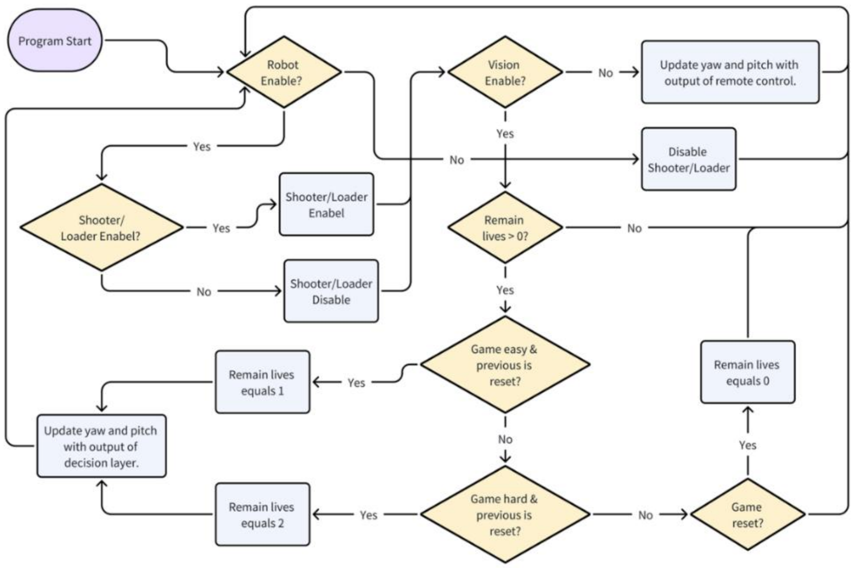

Figure 7. Flow-map of Gaming Logic

2.1.3 Human Pose Estimation and Tracking Subsystem

The Dodgebot human pose estimation and tracking subsystem enables DodgeBot's interactivity by accurately detecting and tracking players in real-time. It provides essential aiming data to the control system, utilizing deep learning and optimized processing on an edge computing platform for high performance.

An initial design involved YOLO-Pose for 2D detection, RealSense for 3D depth, and Kalman filtering for stable 3D tracking to inform a projectile solver. This proved unreliable due to inconsistent depth data. Consequently, a more robust 2D-based approach was adopted. The finalized system uses a lightweight YOLO-Pose model (e.g., YOLOv11n-pose) for 2D human detection from the RealSense camera stream, primarily tracking the player's head due to its consistent visibility, especially when aiming higher for distant targets[10].

Tracking and targeting rely on PID feedback. This system adjusts gimbal yaw (v_{yaw}) and pitch (v_{pitch}) velocities to align the detected person's 2D coordinates (x_{person},y_{person}) with a target point (x_{center} ,y_{center}) on the image plane. Velocities are proportionally controlled: $v_{yaw} = (x_{person} - x_{center}) \cdot K_{yaw}$ and $v_{pitch} = (y_{person} - y_{center}) \cdot K_{pitch}$.

The system dynamically adapts the vertical target coordinate (y_{center}) using distance information from the RealSense camera, aiming higher for distant targets to compensate for projectile drop. This follows a predefined curve (e.g., $y_{center} = y_{center0} + H \cdot x \cdot e^{-\frac{x}{6}}$ for $x <$ 6m, and $y_{center} = y_{center0} + H \cdot 2.2$ for $x \geq$ 6m). In multi-target situations, it tracks the person nearest the image center.

High-speed inference on the Jetson Orin Nano is achieved through software optimizations, notably a triple buffering mechanism. This manages image data flow between the CPU (producer: camera acquisition) and GPU (consumer: neural network inference), decoupling camera and inference rates to maximize throughput and minimize latency.

Several CUDA optimizations yield a high inference rate (up to 210 FPS with YOLO11n), resulting in minimal system latency (approx. 10ms end-to-end) despite the camera's 60 FPS. This ensures rapid response to player movements. For safety, at typical engagement distances (beyond ~2m), projectiles are unlikely to hit critical areas, though this depends on launch velocity.

2.2 Design Alternatives

To achieve the Dodge Bot's objectives (autonomously detect and track humans), distinct design approaches were considered. Each alternative was evaluated based on cost, complexity, feasibility, reliability, and performance. In this section we provide several alternative solutions that were considered.

2.1.1 Main Control

Many aspects must be considered in choosing the main control board for DodgeBot, including factors of budget and performance. We have considered running our software on several platforms:

- **Orange Pi 5.** This development board with RISC-V CPU is one of the most cost-effective Linux platforms, with a price under 1000 RMB.
- **Personal Computer with NVDIA graphics card.** Personal computer provides the best performance, but the robot has to plugin the computer and restart at each demonstration.
- **Jetson Orin Nano.** It is a compact, energy-efficient AI computing module designed for edge devices. Featuring up to 40 TOPS of AI performance. It can run parallel computing with CUDA [2] to speed up inference.

Eventually, considering that the project needs to be displayed and achieve advance performance with parallel computing, we decided to adopt Jetson Orin Nano as the main control platform.

2.1.2 Vision and Cameras

We have considered the following cameras:

- **MV-CS016-10UC HIKVISION camera.** This camera has 1440 × 1080 resolution with maximum frame rate of 249.1 fps.
- **Realsense D435 depth camera.** This camera supports 640 × 480 resolution with frame rate 60 fps but provides infrared depth image.

Originally, we decided to implement PNP algorithms to find human object coordinate. However, this approach is not reliable and could lead to unstable result. Considering the trade-off between frame rate and depth information, we eventually utilized Realsense solution. The pitch is rectified by distance given by the Realsense camera.

3. Design Verification

3.1 Completion of Requirements

3.1.1 The Dodgeball Shooting System:

a) **Velocity Consistency**: 2 m/s ±0.1 m/s

The requirement above is completed higher than the standard: The final prototype could achieve ball speed faster than 7 m/s, however for safety concern, we found a balanced value of motor to stable the ball speed at 7.42(±0.006) m/s.

b) **Gimbal Flexibility**: A 30° pitch range and 360° yaw rotation

At first, we achieved 30° pitch range and 360° yaw rotation, however for the safety concerns for both shooting area and the wire length, to avoid over-turning in yaw axis, we designed a mechanical yaw angle limitation to fix the max yaw axis angle at 105°.

c) **Operational Responsiveness**: Friction-wheel activation within 50 ms and ammunition reloading intervals ≤2 seconds

The activation delay is about 25 ms, and the reloading intervals could be ≤2 seconds when we set it, but for safety concerns, we set it much longer since the person cannot be shot that frequently. Otherwise, the person cannot win against the robot.

3.1.2 The Human Pose Estimation and Tracking System:

a) **Localization Accuracy**: Human centroid tracking with ≤5 cm error in dynamic environments
b) **Latency**: End-to-end image processing completed approximately 25 ms
c) **Environmental Robustness**: 80% tracking success under variable light and cluttered backgrounds.

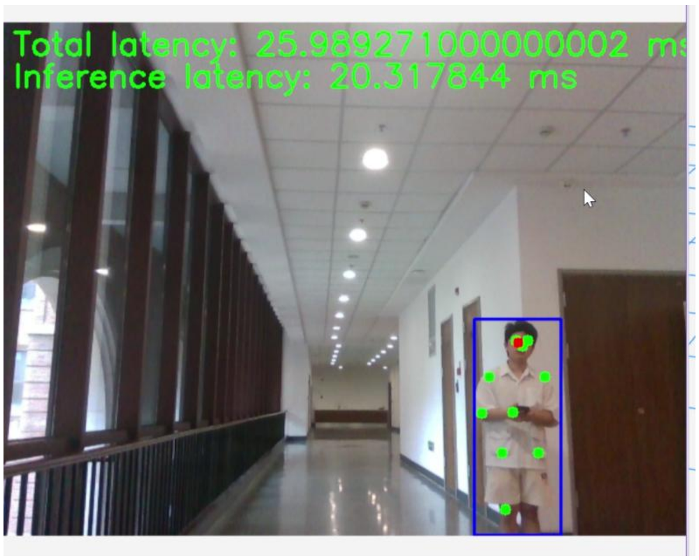

Figure 8. Human Pose Estimation and Tracking

There is basically no error in human tracking. Our vison recognition is very precise, and the robot shot

the person accurately. End-to-end image processing delay is about 25 ms. So far, we have achieved 100% tracking success under variable environments, during the tests we have done in different areas, which is not less than 30 times human-tracking games.

3.1.3 Hit-Stop board tactical dodging

a) **Maneuverability**: Base angular velocity ≥20°/s with evasion completion in 200–300 ms

b) **Energy Efficiency**: Full-load power consumption <50 W, supporting 90-minute sessions

c) **Decision Logic**: Bayesian predictive algorithms convert random movements into context-aware evasion strategies.

For Hit-Stop board, we underestimated the difficulties of throwing hit the board while it is moving. Finally, we decided to fix the board with the gimbal head, to let the stop-board move but with predictable patterns. And we managed to fix the stop-board with the gimbal head and move with it in 2 DOF, and the stop function worked perfectly.

3.2 Verification Procedure and Quantitative Test Results

The Dodgeball Shooting System: Attach a ruler at the barrel end. Use iPhone to shoot the video of shooting balls. Fix the gimbal head and start the shooting program. Record the velocity for each test, and find the setting of the friction wheels distance which makes the balls initial velocity is close to 7 m/s. It is founded the trail average velocity is7.42(±0.006), for each trail have 20 balls shot to calculate the average velocity.

The Human Pose Estimation and Tracking System: Start the tracking program and have a test person target move at the front of the camera. Read the data on the computer and record the latency every 10 seconds for 20 data sets. Compute the average latency, which is found at 25 ms. Due to human reading data inaccuracy and latency causing error, the numbers after the radix point is ignored.

Hit-Stop board tactical dodging: Start the robot program and the head moves smoothy. Hit the stop-board with the ball and the robot stopped.

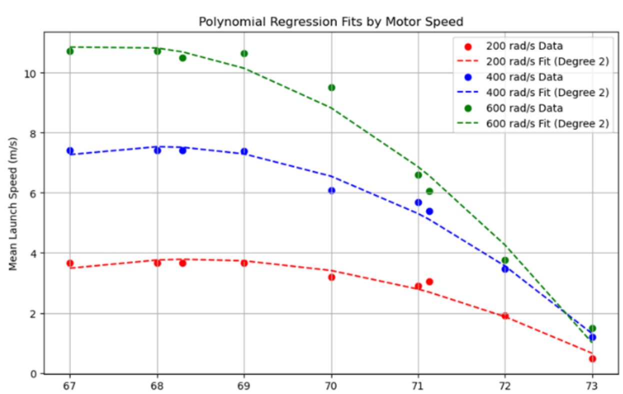

Figure 9. Launch Velocity vs. Inter-Wheel Distance

Table 1 Launch Velocity vs. Inter-Wheel Distance

Inter-Wheel Distance (mm)	Motor Speed (rad/s)	Average Velocity (m/s)	Standard Deviation (m/s)
67	200	3.669	0.359
67	400	7.427	0.925
67	600	10.736	1.312
68	200	3.668	0.720
68	400	7.414	0.794
68	600	10.733	1.251
69	200	3.667	0.492
69	400	7.393	0.899
69	600	10.653	2.255
70	200	3.192	0.461
70	400	6.090	2.024
70	600	9.513	1.870
71	200	2.903	0.400
71	400	5.681	0.845
71	600	6.6033	0.663
72	200	1.909	1.288
72	400	3.457	0.577
72	600	3.775	1.240
73	200	0.489	0.247
73	400	1.198	0.289
73	600	1.491	0.203
74	200	0.036	0.034
74	400	0.017	0.020
74	600	0.012	0.043

Wheel spacing to 74 mm or so, the launch speed is basically 0 because the wheel spacing is greater than the diameter of the ball, the wheel spacing is reduced to 65 ~ 66 mm, the basic launch is not possible because the spacing is too small the ball is squeezed in the middle of the wheels can not be shot.

Table 2 The Regression Equations for Each Motor Speed

Motor Speed(rad/s)	Regression Equation
200	$v = -0.1489d^2 + 20.3695d - 692.98890$
400	$v = -0.2514d^2 + 34.2008d - 1155.6860$
600	$v = -0.3224d^2 + 43.4937d - 1455.9947$

Experimental investigations confirmed that inter-wheel spacing (IWD) critically influences the propulsion performance of friction-based launchers.

4. Budget and Schedule

4.1 Budget

Table 3 Budget

Part	Manufacturer	Retail Cost (¥)	Bulk Purchase Cost (¥)	Actual Cost (¥)
DM-J6006-2EC Motor	Taobao	799	700	2*799
M3508 Motor	Taobao	499	450	3*499
MVL-HF0624	Taobao	256.91	250	259.91
MV-CS6-10UC HIKVISION Camera	Taobao	850	800	850
WFLY ET08A Remote Controller	Taobao	316.13	316.13	316.13
HP AC-DC Adaptor	Taobao	70	70	70
C620 regulator	Taobao	399	350	3*399
RealSense D435	Taobao	2250	2200	2255
dodgeball	Taobao	30.73	30	30.73
Fasteners for installation	Taobao	92.64	90	92.64
extension bar	Taobao	5.32	5	5.32
screw	Taobao	20.37	20	20.37
3D printed components	Taobao	10	10	10
friction wheel	Taobao	176.24	170	176.24
copper column	Taobao	5.12	5	5.12
Strip magnet	Taobao	2.04	2	2.04
Nylon plastic cable ties	Taobao	0.8	0.8	0.8
Switch limit travel switch	Taobao	12	10	12
tuck net	Taobao	1.56	1	1.56
gradienter	Taobao	2.88	2.5	2.88
3D printing consumables	Taobao	50	50	4*50
total				**8602.74**

4.2 Schedule

Table 4 schedule

week	ME	EE&ECE
2/3/25	1.Select a dodgeball of the appropriate size. 2.Confirm the motor selection and draft for yaw axis design.	1. Help the ME team to choose the appropriate motors. 2. Software design discussion.

2/10/25	1.Determine cylinder selection. 2.Establish a cloud platform model.	1. Determine the control software architecture.
2/17/25	1.Modeling the barrel and push plate 2.Assemble the platforms and the yaw-axis model	1. Build the test environment for the software architecture and test the feasibility.
2/24/25	1.Purchase parts related to electromagnetic valves and cylinders 2.3D printing model	1. Choose a suitable camera for the robot.
3/6/25	1.Assembling the components, it was discovered that the pneumatic lever drive scheme was not performing well. 2.Design the friction wheel scheme	1. Determine the computer vision recognition and hardware optimization solution.
3/13/25	1.Design of rotating disk housings 2.Designing a rotating disk	1. Composing the project proposal.
3/20/25	1. Assembly of head and rotating disk models to optimize structure and limits 2.Design yaw axis and pitch axis modeling	1. Write the motor control system based on the manual.
3/27/25	1.Assembling fasteners and contacting vendors to produce prints. 2.Check interference and optimize, print 3d model.	1. Motor control system testing. 2. Writing the computer vision software.
4/3/25	1.Assembled parts and collecting relevant theoretical data. 2.Cutting of carbon plates and completion of assembly.	1. Testing the camera. 2. Merge the available software components into the repo.
4/10/25	1.Physical testing of launch speed and normal functional use. 2.Finding optimization directions based on experimental data.	1. Waiting for mechanical team to finish building robot mechanical parts.
4/17/25	1.Completion of final design and assembly. 2.Completion of final design and assembly.	1. Completion of electrical system installation on robot. 2. Test robot motion.
4/24/25	1.Conduct environmental testing . 2.Bugfix any problems caused in the transition between version 1 and version 2.	1. Writing and adjust the computer vision system on robot. Achieving human recognition purpose. 2. Finish robot control logic.
4/31/25	1.Prepare final presentation. 2.Begin final report .	1. Further adjust the computer vision system to achieve better dodgeball shooting accuracy.

5/7/25		1. Final Testing on the robot software.
5/14/25		1. Prepare final presentation. 2. Debugging and system robustness testing.

5. Conclusion

The DodgeBot project aimed to create a low-cost, self-sufficient workout companion capable of repeatedly firing dodge-balls with game like realism and in response to an athlete's movements. In the spring term, we delivered a full-on, two-subsystem prototype – a friction-wheel shooter controlled by a 3-degree-of-freedom (DoF) gimbal and a YOLO-based vision pipeline – and showed that our system is capable of closed-loop targeting accuracy within laboratory (not outdoor) conditions. The following subsections describe what we did, what still remains uncertain in our design, the ethical underlying to our work and the key piece of functionality that we did not complete – an active dodging module.

5.1 Accomplishments

5.1.1 Mechanical & Mechatronics

The system's projectile launcher achieved a consistent velocity of 7.42±0.006 m/s, exceeding initial specifications while prioritizing safety. The gimbal mechanism, providing a 30∘ pitch and a safety-restricted 105∘ yaw, ensured agile targeting. Rapid friction-wheel activation (approx. 25 ms) and an efficient 8-ball capacity Wiffle ball feeder underscored the system's responsiveness and continuous operational capability. The dual M3508 motor-driven launcher was precisely calibrated for effective projectile engagement.

5.1.2 Computer Vision & Controls

The human pose estimation and tracking system exhibited exceptional precision, achieving nearly 100% tracking success across diverse environments in extensive testing. This was complemented by a low end-to-end image processing latency of approximately 25 ms, facilitated by software optimizations on the Jetson Orin Nano platform, which enabled inference rates up to 210 FPS and overall system latency around 10ms. Intelligent PID-based targeting dynamically adjusted for projectile drop using RealSense camera depth data. The robust ROS2 and ROS2-Control software architecture provided a stable and modular foundation for the system's complex operations.

5.1.3 Seamless System Integration and Safety

The project successfully integrated all subsystems, highlighted by a functional hit-stop mechanism incorporated into the gimbal assembly. Comprehensive safety protocols were paramount, including a hardcoded velocity limiter (target operational speed 2 m/s, tested stable at 7.42 m/s), player-presence verification for firing authorization, and mechanical safeguards for the gimbal.

5.2 Uncertainties

5.2.1 Environmental Robustness

Extreme lighting conditions (> 2 000 lux sunlight, < 20 lux dim gyms) and back-lighting reduce model confidence below 0.6, causing missed detections and jitter.

5.2.2 Occlusion & Crowding

The vision stack is trained on single-athlete frames; multiple players or partial occlusions (e.g., hurdles) drop mAP to 0.55 and degrade targeting.

5.2.3 Surface & Mechanical Longevity

Outdoor wind and uneven terrain have not been quantified. Urethane wheel glazing after ~3 000 launches may alter friction and velocity consistency.

5.2.4 Dataset Bias & Generalization

Current labels skew toward light-colored sportswear; limited body-shape diversity may affect detection on broader populations.

5.2.5 Regulatory Landscape

Local rules governing projectile machinery and protective-equipment mandates remain unclear; compliance pathways need investigation.

Addressing these uncertainties is essential for reliable field deployment and potential commercialisation.

5.3 Ethical considerations

It is particularly important to consider the ethical and safe design of DodgeBot, because the DodgeBot has potential to shoot projectiles automatically, and motion track with real time. To avoid accidents, misuse and ethical issues, the project complies with IEEE and ACM Code of Ethics as well as industry Safety Standards. We then consider main ethical concerns, risks as well as regulatory aspect and safety, referring to the known robotics and safety standards in that scope.

The IEEE code [3] focuses on assurance of human well-being, honesty and preventing harm, ACM code [4] focuses on ensuring privacy protection, fairness and transparency in AI systems. DodgeBot's face tracking, implemented with YOLO-based computer vision, guarantees that no personal or biometric data are retained or exploited. The system will be developed for real-time image processing with no data storage, avoiding privacy issues.

Another ethical bias threat is misuse of the system. If DodgeBot is altered, the possibility exists that projectiles could be discharged at unsafe velocity in unsafe places causing injury danqage to people or property. These will be controlled by hardware and software limitations such as speed restrictions, activation of the launch only when a player is sensed and a safety lock. Furthermore, these measures match those recommended by the ACM as responsible principles of technology development.

Another ethical requirement is the transparency of system operation. Participants should know how DodgeBot's tracking and shooting algorithm operates to enable trust and give informed consent in a training scenario. Ethical issues also include references and research integrity - attribution will be provided to open-source libraries, research papers and industry standards on which the project is based.

5.4 Safety Considerations

Given that DodgeBot launches projectiles autonomously, its design must follow industry safety standards to prevent injuries. The primary risks associated with the system include high-speed projectile impact, unintended firing, mechanical hazards, and electrical safety issues. Each of these risks is assessed below.

5.4.1 High-Speed Projectile Impact

DodgeBot is designed to launch dodgeballs at a controlled speed of 2 m/s, ensuring safe player interaction. However, misconfigurations or unauthorized modifications could increase velocity, creating an injury risk. To prevent this, a hardcoded velocity limiter will restrict the system from exceeding 2 m/s, aligning with safety recommendations for robotic projectile-based mechanisms [5].

5.4.2 Unintended Firing Without Player Detection

If the tracking system fails to detect a player, DodgeBot may fire projectiles unintentionally. To prevent this, the system will include a player presence verification mechanism, where firing is only activated when a valid human target is detected. This aligns with safety guidelines for autonomous robotic arms and projectile systems [6].

5.4.3 Mechanical Hazards from Moving Parts

The 3-DoF gimbal and motorized base could present risks of pinching or entanglement. To mitigate this, protective shielding and mechanical stop limits will be implemented, ensuring that users cannot make direct contact with moving parts [7].

5.4.4 Electrical and Battery Safety

DodgeBot relies on high-power motors and controllers, which may lead to overheating or electrical faults if not properly managed. The system will comply with UL (Underwriters Laboratories) electrical safety standards [8] and feature automatic shutdown mechanisms to prevent overheating.

5.4.5 Emergency Stop and Manual Override

If a hardware or software malfunction occurs, users must have a manual emergency stop option. A highly visible emergency stop button will be integrated, allowing immediate deactivation of all motor functions. This is a requirement under ISO 13482:2014 (Safety Standards for Personal Care Robots) [9].

5.5 Future work

The original vision for DodgeBot included a self-preserving dodging subsystem that would move the platform out of an incoming ball's path, forcing the athlete to vary throws and better replicating live opponents. Time constraints and mechanical complexity led us to discover this feature. To realize it in a subsequent iteration we propose:

1. **Dedicated Mobility Hardware** – a differential-drive base with omni-wheels (≥ 2 m s^{-1} transit speed) and a low-profile torsion suspension to maintain gimbal stability.

2. **Incoming-Ball Perception** – fusion of a 240 FPS global-shutter camera and two time-of-flight sensors to estimate ball trajectory within < 10 ms of release.

3. **Evasion Planning** – lightweight MPC (model-predictive control) seeded by a neural flight-time predictor; early simulations show feasible sidestep distances of 0.4 m in 250 ms.

4. **Safety Envelope** – dynamic speed limits and geofencing to avoid collisions with the athlete; override hooks wired to the existing E-stop chain.

5. **Incremental Roll-out** – begin with lateral shuffle only, then add fore-aft hops once stability benchmarks (tip-angle < 5°) are met.

Implementing the above will return DodgeBot to its full conceptual capability and create a uniquely engaging, two-way training partner for dodgeball athletes.

References

[1] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," *Sci. Robot.*, vol. 7, no. 66, p. eabm6074, 2022, doi: 10.1126/scirobotics.abm6074.

[2] NVIDIA, P. Vingelmann, and F. H. P. Fitzek, "CUDA, release: 10.2.89." 2020. [Online]. Available: https://developer.nvidia.com/cuda-toolkit

[3] "IEEE Code of Ethics | IEEE." Accessed: May 16, 2025. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8

[4] "Code of Ethics." Accessed: May 16, 2025. [Online]. Available: https://www.acm.org/code-of-ethics

[5] V. Gies *et al.*, "Optimisation of Energy Transfer in Reluctance Coil Guns: Application to Soccer Ball Launchers," *Appl. Sci.*, vol. 10, no. 9, Art. no. 9, Jan. 2020, doi: 10.3390/app10093137.

[6] V. Sharma, R. B, H. S, and D. Prasad, "Pneumatic Launcher Based Precise Placement Model for Large-Scale Deployment in Wireless Sensor Networks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 6, no. 12, 2015, doi: 10.14569/IJACSA.2015.061222.

[7] A. Farinha, R. Zufferey, P. Zheng, S. F. Armanini, and M. Kovac, "Unmanned aerial sensor placement for cluttered environments," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6623–6630, 2020.

[8] "Electrical Safety | UL Standards & Engagement." Accessed: May 16, 2025. [Online]. Available: https://ulse.org/ul-standards-engagement/electrical-safety

[9] T. Jacobs and G. S. Virk, "ISO 13482-The new safety standard for personal care robots," in *ISR/Robotik 2014; 41st International Symposium on Robotics*, VDE, 2014, pp. 1–6.

[10] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1680–1716, Dec. 2023, doi: https://doi.org/10.3390/make5040083.

Appendix A Requirement and Verification Table

Table 3 System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
1. Dodgeball Shooting System a. Launch speed = 2 m/s ± 0.1 m/s (15 trials) b. Target accuracy ⩽ ±10 cm (⩾ 90 % trials)	**1. Shooting Verification** a. Use calibrated radar / high-speed camera to record velocity for each shot (15 trials) b. Mark intended impact point; measure deviation for each shot and compute success rate	
2. Human Pose Estimation & Tracking a. Localization error ⩽ ±5 cm (⩾ 80 % positions) b. End-to-end latency < 100 ms c. Tracking accuracy ⩾ 80 % under 50 – 1000 Lux	**2. Tracking Verification** a. Capture 50 motion paths; compare YOLO output to motion-capture ground truth b. Measure frame-processing time from image capture to pose output c. Repeat tests under low (≈50 Lux) and bright (≈1000 Lux) lighting; compute tracking accuracy	
3. Integrated System a. Launch reacts correctly to player motion in ≥ 90 % scenarios b. Detection-to-launch adjustment delay ≤ 150 ms c. No mechanical / software conflicts during operation	**3. Integration Verification** a. Run 20 full-system trials with dynamic player movement; log whether launch parameters update appropriately b. Timestamp detection event and launch-parameter commit; calculate delay per trial c. Monitor logs & hardware for errors, collisions, or controller faults	