

# AI Camera Tracking System Report

*Utkrisht Batra, Coree Joseph, Alexander Leon*

*Electromechanical Engineering Technology Department*

*College of Engineering*

*California State Polytechnic University, Pomona*



## Table of Contents

|      |                                                           |    |
|------|-----------------------------------------------------------|----|
| I.   | Introduction .....                                        | 4  |
| II.  | Problem Statement .....                                   | 5  |
| A.   | Specifications and Constraints .....                      | 5  |
| B.   | Cloud-Based System Vulnerabilities .....                  | 5  |
| III. | Design Solution .....                                     | 6  |
| A.   | Key Features and Operation .....                          | 8  |
| IV.  | Design Verification .....                                 | 12 |
| A.   | Measured Automated Validation .....                       | 12 |
| B.   | Simulation Support .....                                  | 13 |
| C.   | Pending Live Validation .....                             | 15 |
| V.   | Approach to Solution .....                                | 16 |
| VI.  | Conclusions and Recommendations .....                     | 17 |
|      | Appendix A - CAD Drawings .....                           | 18 |
|      | Appendix B – Electrical / Assembly Support Material ..... | 24 |
|      | Appendix C – Software / Prototype Code .....              | 25 |
|      | Appendix D – Cost / Vendor Information .....              | 27 |
|      | Appendix E – Future Work .....                            | 28 |
|      | References .....                                          | 29 |

***Abstract***—This project presents the design and implementation of a dual-axis AI-powered pan-and-tilt system capable of real-time human target detection and face-preferred tracking. The system integrates an NVIDIA Jetson Orin Nano for high-performance computing with an OpenCV-based software architecture to support low-latency target centering. A PCA9685 servo driver manages the pan-and-tilt mechanics, using high-torque servos to support smooth and responsive physical tracking. Hardware components such as the camera cover were developed using SolidWorks for mechanical integration and structural support. Initial validation demonstrated successful onboard detection, software-limited pan/tilt control, local logging, and stable bench-level target tracking under controlled conditions, providing a scalable foundation for future BillyBOT integration.

***Keywords***—*AI tracking, pan-tilt system, Jetson Orin Nano, OpenCV, embedded vision, robotics integration*

## I. INTRODUCTION

The AI Camera Tracking System is a collaborative engineering project developed to give the BillyBOT campus guide platform autonomous person-tracking capability. The project was driven by the need for a private, low-latency camera system that can operate independently of internet connectivity and cloud-based storage. The main goals were to implement local tracking, protect the mechanism through software-defined safety limits, and create a modular platform that future students can continue to improve. This report presents the design requirements, system implementation, and current verification status of the prototype.

The following sections describe the problem constraints, the selected hardware and software architecture, and the current results from automated validation and simulation-supported analysis. Together, these sections show how the project moved from an early unstable prototype into a functioning, demonstrable tracking system with a repaired camera path, repeatable logging, and a defined validation workflow.

More broadly, the project was intended to address several common limitations of modern commercial camera systems, including cloud dependency, internet-related failure points, subscription costs, and reduced flexibility for robotic integration [5], [6], [7], [9]. By keeping storage and processing on the device, the system avoids relying on off-site services and remains easier to adapt for custom use. The design also stays modular, making it practical for student development, hardware revision, and future upgrades. At its current stage, the prototype has demonstrated credible bench-level performance and is now in the structured validation and performance-refinement phase before fuller BillyBOT integration.

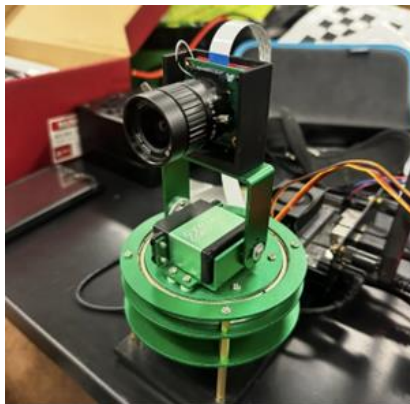


Fig. 1. Integrated AI camera tracking prototype in the bench testing environment.

## II. PROBLEM STATEMENT

Many commercial security systems depend on off-site servers for video storage, viewing, or processing, which introduces communication delay, network dependency, and added security risk [5], [6], [9]. These systems are also vulnerable to Wi-Fi jamming or deauthentication attacks that can interrupt operation during critical events [7]. In addition, most off-the-shelf products do not offer the modularity or control needed for direct integration into a mobile robotic platform such as BillyBOT.

### A. Specifications and Constraints

- **Offline Operation:** All vision inference and tracking logic must run locally on the Jetson Orin Nano.
- **Targeting:** The system must detect human targets using an OpenCV DNN-based pipeline with fallback behavior when a full face or ideal target view is not available.
- **Safety Envelope:** Software must restrict commanded motion to pan  $60^{\circ}$ – $120^{\circ}$  and tilt  $30^{\circ}$ – $130^{\circ}$  to protect the mechanism and wiring.
- **Storage:** Local NVMe storage is required for evidence of logging without cloud dependence.
- **Power:** The system must use regulated DC power capable of supporting the Jetson processing load and servo demand during operation.
- **Latency Target:** The system should maintain a practical end-to-end tracking response target of approximately 200 ms or better under controlled conditions.

### B. Cloud-Based System Vulnerabilities

Consumer camera systems such as Ring, Wyze, and Nest often rely heavily on cloud-based workflows for storing and retrieving footage [5], [6], [9]. This dependence creates network-related failure points and makes the system more vulnerable to disruption. In wireless systems, a jammer or deauthentication attack can interfere with connectivity and reduce the usefulness of the camera during an event [7]. These systems can also introduce additional communication and retrieval delays compared with local processing architecture [6], [9].

Wired systems reduce some of these problems, but they are often less flexible and are not always practical for modular robotic integration. For this project, the main limitation was not simply whether a system was wired or wireless, but whether it could operate independently, store data locally, and be adapted for BillyBOT's architecture.

### III. DESIGN SOLUTION

The final design is a modular, self-contained tracking module built around the NVIDIA Jetson Orin Nano. Video is provided through the Arducam IMX477 HQ camera using a repaired and verified CSI-2 / Argus path [3], [10]. Mechanical actuation is handled by a Yahboom 2-DoF metal pan-tilt assembly. This hardware provides a wide physical range of motion, with approximately  $270^\circ$  of raw pan travel and  $180^\circ$  of raw tilt travel, but automated operation is intentionally restricted to the verified software-safe envelope to protect the mechanism and wiring.

The system is designed to run fully offline and remain easy to modify for future student development. The IMX477 camera provides high-quality imaging and automatic IR-cut functionality for day and night operation, making it suitable for mixed-light conditions [10]. When mounted onto BillyBOT, the module is intended to operate from BillyBOT's onboard power system through appropriate regulated power conversion for the processing and actuation hardware.

The software pipeline uses OpenCV DNN-based detection and tracking logic to identify a target and convert pixel error from the frame center into coordinated pan and tilt commands [2]. These commands are transmitted over the I2C bus to the PCA9685 PWM driver, which controls the pan-tilt servos [4]. In the current implementation, target selection is face-preferred, with fallback behavior used when a full face is not clearly visible. Local NVMe storage is used to preserve captured data and log outputs directly on the device without relying on cloud services.

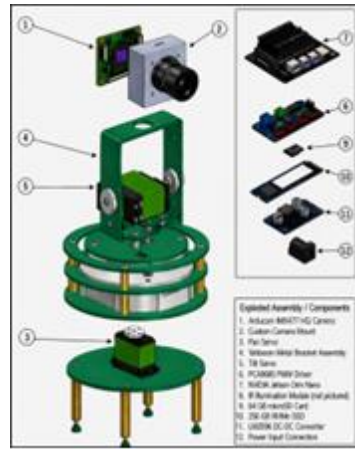


Fig. 2. Assembly Exploded.

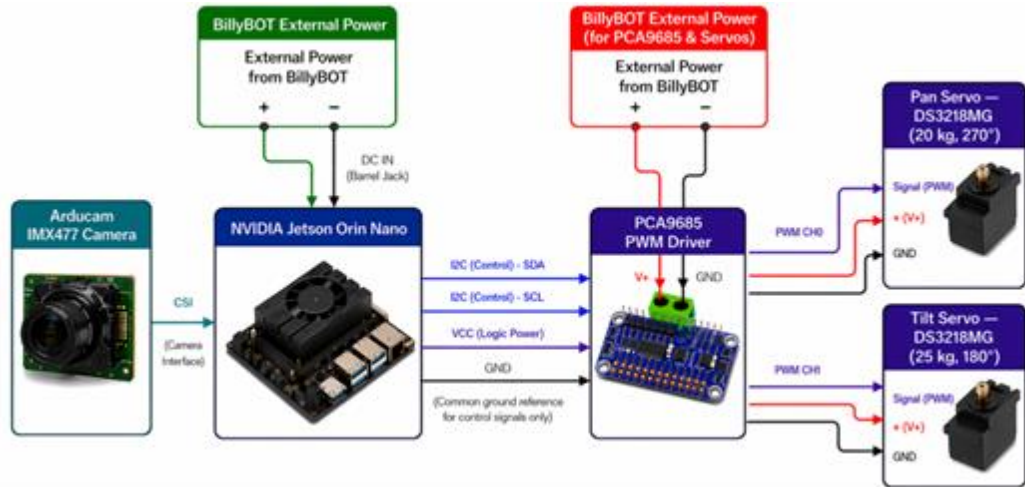


Fig. 3. Electrical Design Schematic.

## Electrical Schematic Explanation

**Camera Interface:** The Arducam IMX477 HQ connects to the NVIDIA Jetson Orin Nano through the CSI-2 interface for high-speed image transfer.

**Servo Control:** The PCA9685 PWM driver connects to the Jetson Orin Nano through the I2C bus and generates servo control signals for the Yahboom pan-tilt assembly.

**Power Management:** The system uses regulated DC power conversion sized to support the Jetson processing load and the servo demand during offline operation.

### A. Key Features and Operation

The tracker is designed for calm, deliberate motion using a state-machine architecture. The logic begins with an IDLE\_CENTER condition and transitions into target acquisition. Once a valid target is detected, the system enters TRACK, where it continuously works to keep the target near the center of the frame. If the target is briefly obscured, the tracker can move into HOLD or LOCAL\_REACQUIRE before either regaining lock or returning to center through RETURN\_TO\_CENTER. This structure helps reduce unnecessary motion and improves stability during real operation.

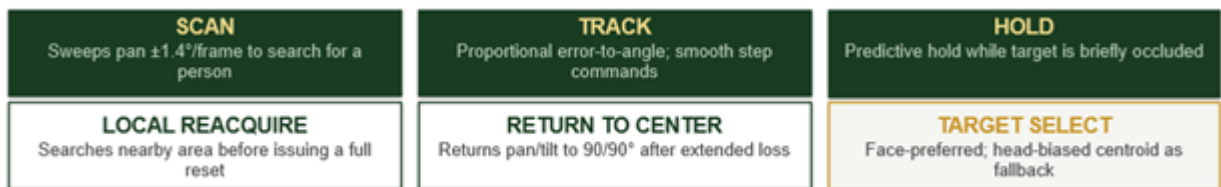
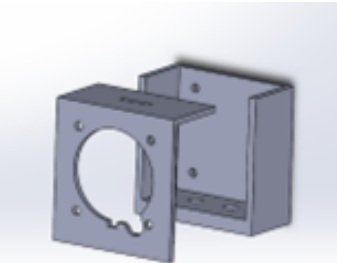


Fig. 4. Tracker State Machine Diagram.

TABLE I. KEY PARTS AND FEATURES

| Parts                        | Purpose                                                                                                                                   | Pictures                                                                              |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| Jetson Orin Nano             | A 6-core ARM CPU and 1,024-core Ampere GPU providing 102 GB/s bandwidth. It serves as the primary processing unit for local AI inference. |  |
| Yahboom 2-DoF metal pan-tilt | A durable metal 2-DoF assembly providing a wide range of pan and tilt motion for target tracking.                                         |  |

|                                 |                                                                                                                           |                                                                                       |
|---------------------------------|---------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| 25 kg and 20 kg DS3218MG Servos | High-torque actuators are designed to provide precise movement and support the mechanical load of the camera assembly.    |    |
| Arducam IMX477 HQ camera        | A 12.3 MP high-resolution camera with an automatic IR-cut filter for consistent imaging in both day and night conditions. |    |
| Camera Mount                    | Custom-designed housing that secures the camera to the pan-tilt kit while protecting internal circuitry.                  |  |
| 64 GB microSD Card              | Removable storage for easy data transport and secondary logging to supplement the onboard SSD.                            |  |

|                        |                                                                                                               |                                                                                      |
|------------------------|---------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| 256 GB NVMe SSD        | Primary high-speed storage for local evidence logging and fast system boot times.                             |   |
| LM2596 DC-DC converter | High-efficiency buck regulator used to step down input voltage to a stable level for the Jetson and servos.   |   |
| PCA9685 module         | A 16-channel PWM driver that controls multiple servos via the I2C bus, minimizing pin usage on the processor. |  |

### Custom piece design:

The only custom-designed part of the system is the camera mount and protective cover used to secure the IMX477 camera to the pan-tilt assembly. This part was developed to support quick iteration, straightforward manufacturing, and better protection of the camera board during handling and operation. Additive manufacturing was selected because it allowed the mount geometry, hole placement, and cable clearances to be produced quickly and revised easily during development.

### Printing specifications:

- Nozzle temperature: 210°C–240°C

- Bed temperature: 50°C–60°C
- Infill: 20%–40% using a triangular pattern for improved stiffness
- Raft: 5–7 total layers when needed for print stability
- Cooling fan: on continuously to reduce warping during printing

### **System Specifications**

- Camera resolution: 12 MP IMX477, with 1080p at 30–60 FPS modes
- Inference performance: variable FPS, with a design target of approximately 15–20 FPS using OpenCV DNN processing
- Mechanical servo travel: 270° pan and 180° tilt as raw physical capability
- Software command update time: approximately 0.12–0.20 s per commanded step during dry-run timing only, without physical feedback confirmation
- Software-safe operating range: pan 60°–120°, tilt 30°–130° for protection of the mechanical assembly during automated tracking
- Power requirement: regulated DC power sized to support both Jetson processing load and servo demand during operation
- Tracking latency target:  $\leq 200$  ms end-to-end

### **Programming Summary:**

The software stack uses Python and OpenCV for image capture, preprocessing, and DNN-based inference, with state-machine logic used for tracker behavior and motion control. Video data, logs, and supporting outputs are written locally to NVMe storage. Additional command-line utilities are used to verify servo endpoints and deadband settings during setup. During operation, the tracker converts the target position relative to the frame center into coordinated pan and tilt updates. Extensive troubleshooting was required to calibrate the control loop. Early iterations had direction-sign errors that caused the servos to move away from the subject instead of centering it. After the directional logic was corrected, the proportional gains were tuned to balance responsiveness and stability. The tracking logic was also refined for multi-person scenes, partial obstruction, and different lighting conditions while maintaining lock on the primary subject.

#### IV. DESIGN VERIFICATION

Design verification was performed through measured automated validation, modeled simulation support, and integrated prototype observation on BillyBOT. The baseline validation runs confirmed stable target detection across the sampled sequence, zero recorded target losses, and operation within the software-safe motion boundaries of pan  $60^{\circ}$ – $120^{\circ}$  and tilt  $30^{\circ}$ – $130^{\circ}$  with no boundary clamp hits. Integration-level testing on BillyBOT was also used to evaluate whole-system fitment, cable routing, and clearance between the pan-tilt assembly and the robot frame during motion. At this stage, verification should be understood as a functional subsystem and integration-level confirmation rather than fully closed-out performance validation [12].

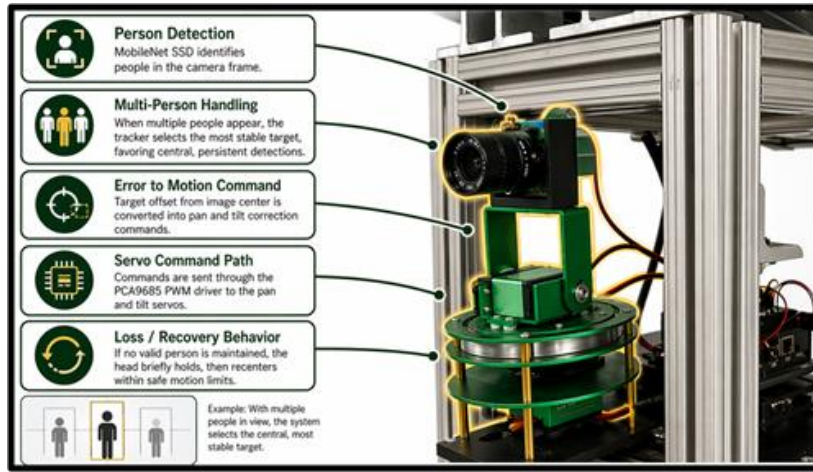


Fig. 5. Labeled photo of AI Camera tracking prototype mounted on BillyBOT.

##### A. Measured Automated Validation

Hardware in the loop validation was performed directly on the NVIDIA Jetson Orin Nano to assess the system's performance under complex, real-world conditions. The baseline automated run achieved a 100% detection rate, recording 10 out of 10 detection frames with 9 successful face target selections and zero target losses. During in-person testing, the OpenCV-based state machine demonstrated reliable behavior by reacquiring the target after momentary obstruction and maintaining lock on the primary subject when secondary individuals crossed the field of view. Real-time monitoring confirmed that the Yahboom 2-DoF mechanism stayed within the software-defined safety boundaries of  $60^{\circ}$ – $120^{\circ}$  pan and  $30^{\circ}$ – $130^{\circ}$  tilt, with zero recorded clamp hits.

Because the system is mounted on top of BillyBOT, the full raw servo range was not needed during automated tracking.



Fig. 6. Representative live tracking frame showing person detection and active target selection.

#### *B. Simulation Support*

The simulation model was used to illustrate the expected behavior of the tracking logic under defined conditions, including target occlusion and multi-person priority handling. To ensure the system was robust, simulations were expanded to include multi-person environments, testing the algorithm's ability to maintain tracking priority on a primary subject even when secondary targets entered the frame. Variable lighting conditions and complex image scenarios were also simulated, such as high contrast backgrounds and low light noise, to determine the threshold at which the detection confidence would drop.

The simulation was also used to check the software-safe motion boundaries of  $60^{\circ}$ – $120^{\circ}$  pan and  $30^{\circ}$ – $130^{\circ}$  tilt so the commanded motion stayed within the protected operating range before live hardware testing. This allowed verification that the code would electronically soft stop the motors before they hit mechanical limits, preventing gear strain or hardware failure. While these results are vital for understanding tracker logic and safety boundary behavior, they serve as a behavioral baseline rather than final closed loop hardware proof.

The target position vs commanded pan response plot is a modeled control-response curve derived from the implemented tracker tuning: 1280 px frame width, center at 640 px, pan deadband 28 px, pan gain 24.5, max pan step 3.2 deg, and pan center 90 deg. Those parameters come from the simulation summary for the presentation demo sequence.

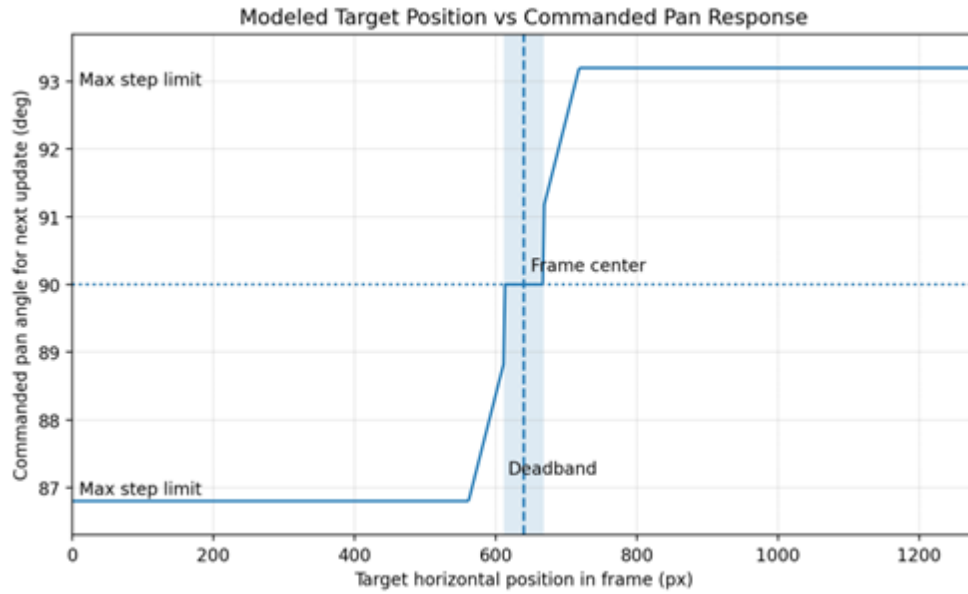


Fig. 7. Target position vs commanded pan response plot.

The safety-boundary behavior plot is based on the available simulation summaries for the presentation of demo and safety-boundary scenarios, using their final commanded pan, final estimated pan, average simulated error, and target-loss counts. It is simulation-summary based, not a live physical measurement.

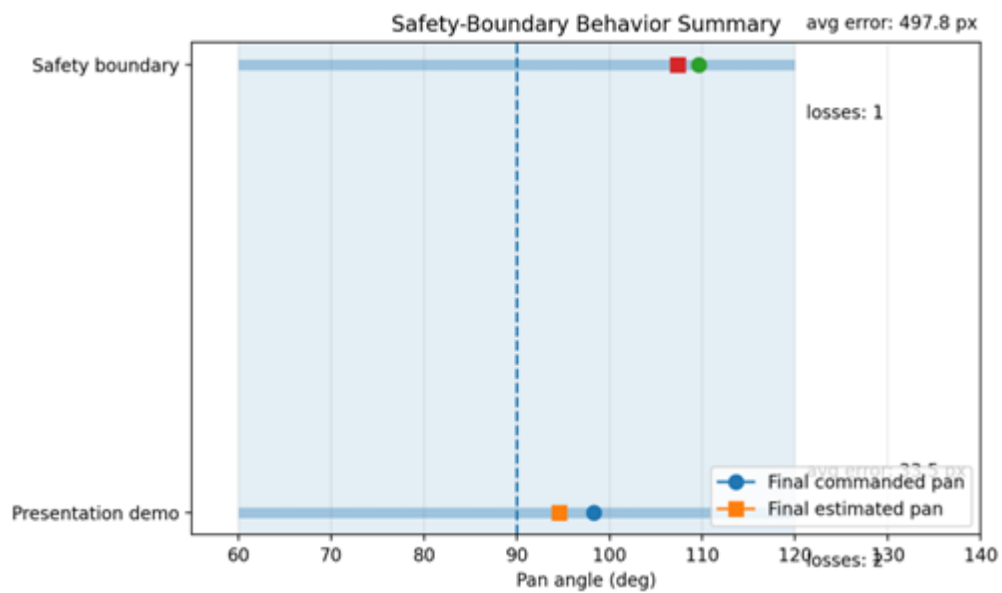


Fig. 8. Safety-boundary behavior plot.

### C. Pending Live Validation

Structured human-in-the-loop live scenarios, including broader multi-person target selection and faster target motion cases, are still limited and require additional structured testing. Functional-level multi-person testing has been performed, but broader repeatable validation is still incomplete.

TABLE II. BASELINE VALIDATION DRY-RUN METRICS

| Metric                       | Result |
|------------------------------|--------|
| Startup success              | Yes    |
| Camera opened                | Yes    |
| Detector loaded              | Yes    |
| Average FPS                  | 3.11   |
| Frames processed             | 10     |
| Frames with detections       | 10     |
| Face-based target selections | 9      |
| Pan clamp hits               | 0      |
| Tilt clamp hits              | 0      |
| Target losses                | 0      |
| Successful reacquisitions    | 0      |
| Target switches              | 0      |

### Current Verification Limits

Current verification is still limited in several important areas and will be expanded through additional live testing. Servo timing results currently represent software command dispatch and update timing only. Because the system does not include encoder feedback, these values should not be interpreted as true physical servo settling time. Functional-level multi-person testing has been performed, but broader structured validation is still limited. Additional follow-up, robustness, and polished demonstration runs are still needed to fully characterize real-world behavior. The available evidence also does not yet close out all final quantitative requirements, including formal end-to-end latency and target-centering accuracy. The measured 3.11 FPS result was obtained during a dry-run baseline and should not be treated as the final optimized system performance.

### Requirements vs. Current Verification Status

The project status relative to the main design requirements has been updated to reflect the completed baseline validation and additional live testing already performed.

TABLE III. DESIGN REQUIREMENTS VERIFICATION STATUS

| Requirement                       | Current Verification Status                                                                       |
|-----------------------------------|---------------------------------------------------------------------------------------------------|
| Offline/onboard processing        | Verified in baseline run                                                                          |
| Local data storage                | Verified in baseline run                                                                          |
| Pan-tilt actuation response       | Partially verified / Improved with command-path evidence                                          |
| Human target tracking             | Partially verified with automated baseline and working live-follow capability                     |
| Low-light operation               | Pending live test                                                                                 |
| Quantitative latency and accuracy | Partially improved, limited by pending structured live testing and lack of servo encoder feedback |

## V. APPROACH TO SOLUTION

Several design approaches were considered before the final architecture was selected. Cloud-dependent systems were rejected because they introduce external dependency, off-device storage concerns, and vulnerability to network disruption. Fixed-camera designs were also not sufficient for BillyBOT's intended guide function, since the robot requires an active field of view rather than passive observation.

To address these limitations, the team selected an offline, modular architecture centered on the NVIDIA Jetson Orin Nano. This approach supports local vision processing, active target tracking, and a more practical path toward integration with a mobile robotic platform. The final design was chosen because it provided the best overall balance of local processing capability, modularity, serviceability, and future expandability for continued student development.

TABLE IV. TRACKING SYSTEM CONCEPT SELECTION MATRIX

| Concept                        | Offline Operation | Tracking Ability | BillyBOT Integration | Modularity | Result   |
|--------------------------------|-------------------|------------------|----------------------|------------|----------|
| Cloud camera system            | Low               | Medium           | Low                  | Low        | Rejected |
| Fixed local camera             | High              | Low              | Medium               | Medium     | Rejected |
| Offline Jetson pan-tilt system | High              | High             | High                 | High       | Selected |

## VI. CONCLUSIONS AND RECOMMENDATIONS

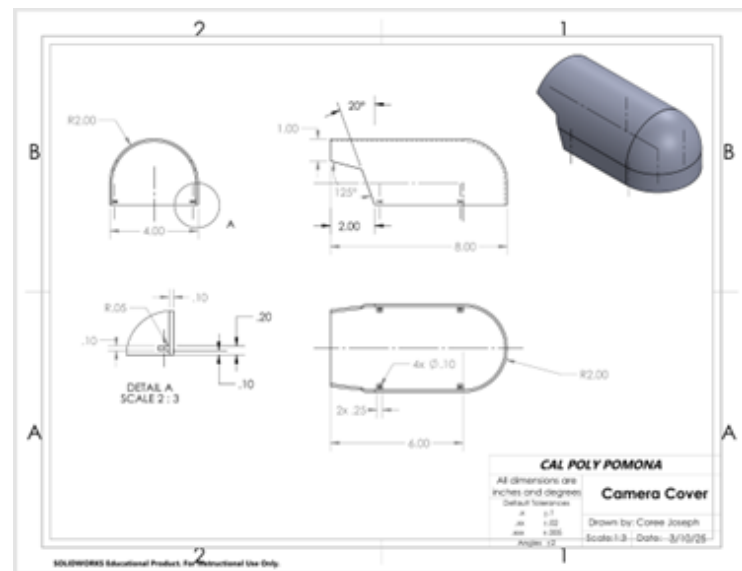
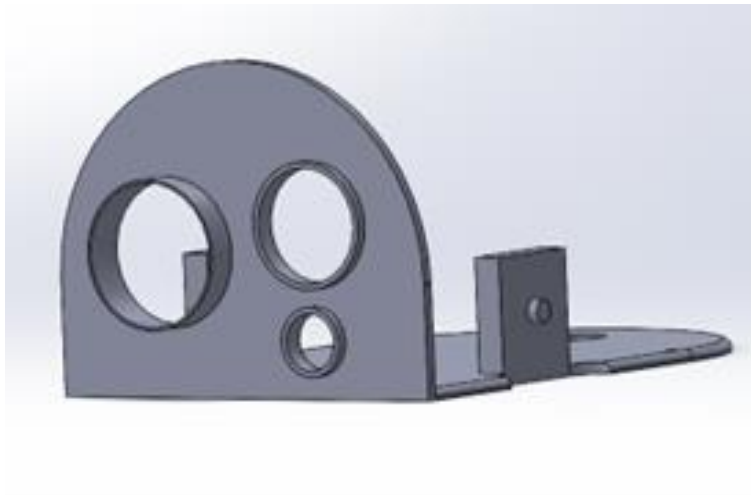
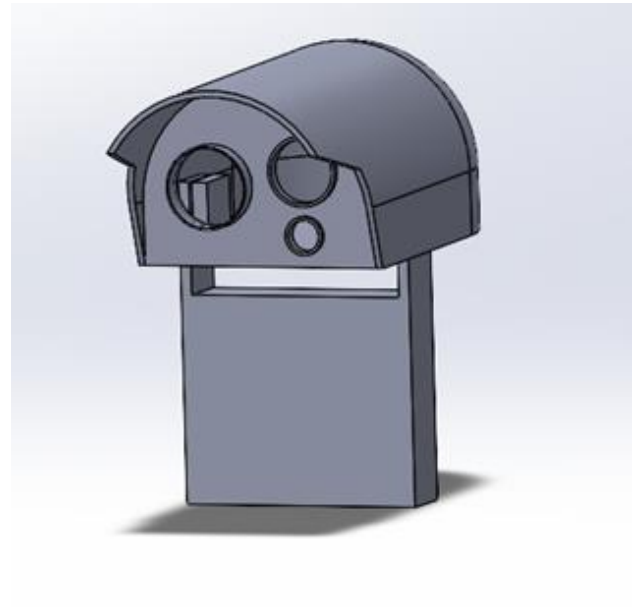
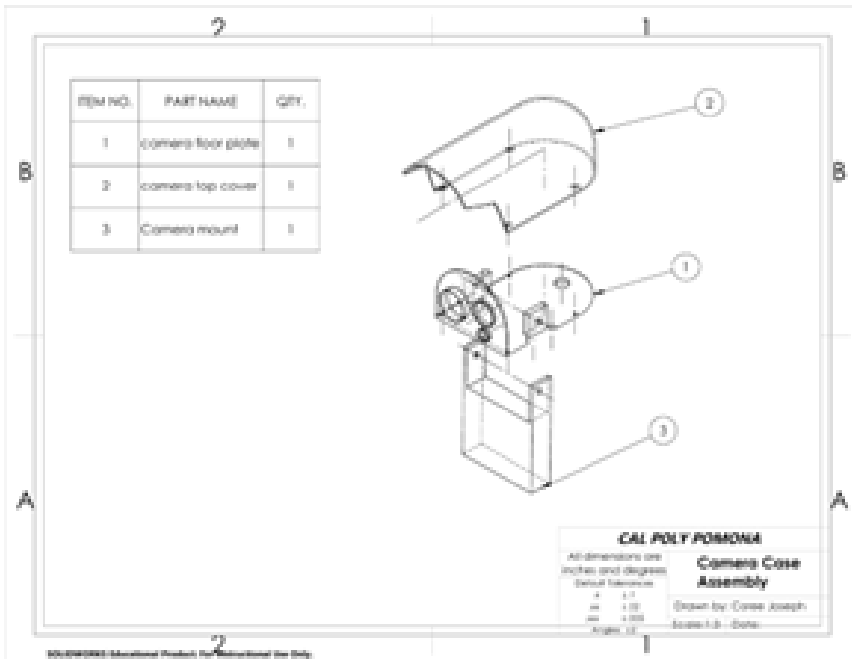
The project successfully moved from concept to a functioning offline AI camera tracking prototype. The final system combines the Jetson Orin Nano, IMX477 HQ camera, pan-tilt assembly, PWM control hardware, and local storage into a working platform that is suitable for bench demonstration and further BillyBOT integration. The repaired camera path, repeatable logging process, and software-safe motion limits gave the project a stable base for continued development.

Current testing supports that the prototype can detect and track a target within the verified software-safe motion envelope under controlled conditions. Mounted testing on BillyBOT also helped evaluate system fitment, cable routing, and general integration behavior. Even so, the project should still be described as functionally demonstrated rather than fully validated, since broader structured live validation and final quantitative characterization are still in progress.

The next steps recommended are to complete the human-in-the-loop validation runbooks, continue improving false-positive rejection and multi-person handling, and further tune the control loop for smoother and more reliable motion. Additional cable routing and final mounted integration cleanup on BillyBOT should also be completed before broader demonstration use. Overall, the project resulted in a credible, modular prototype that future students can continue to refine and expand.

## APPENDIX A - CAD DRAWINGS

Fig. A1. First camera-cover concept and early camera floor plate drawing.



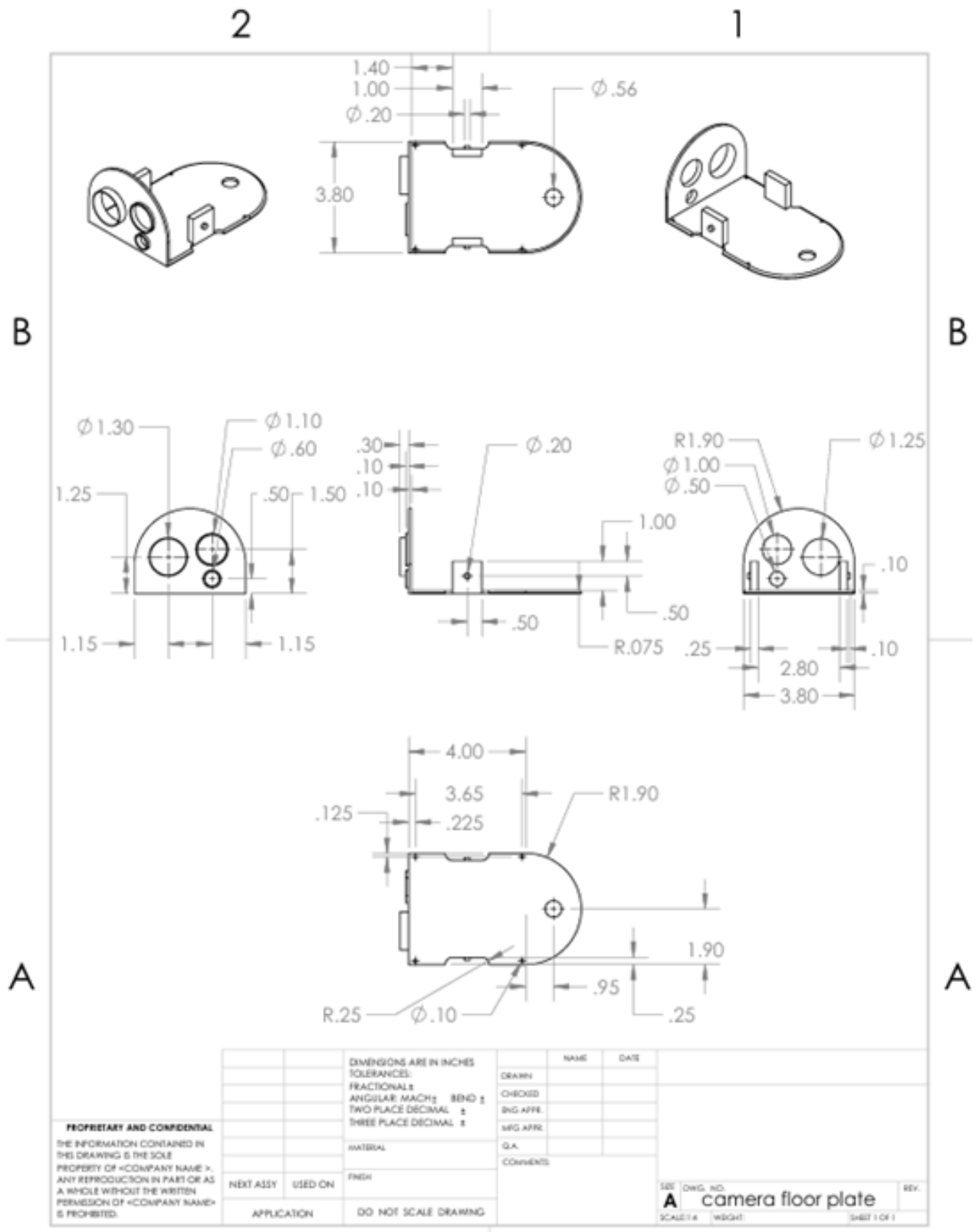


Fig. A2. Second camera-cover design iteration.

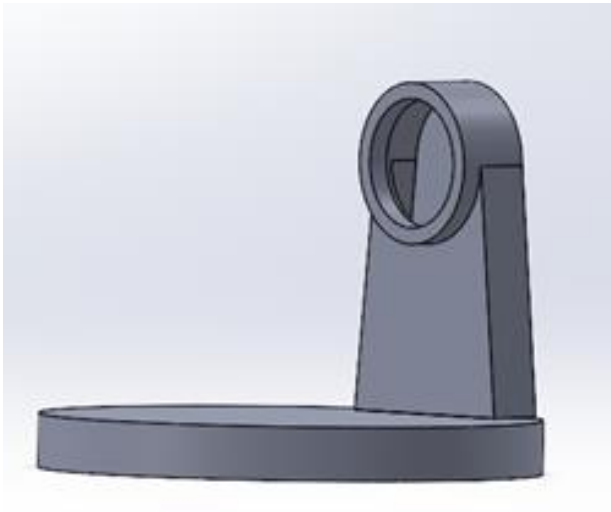
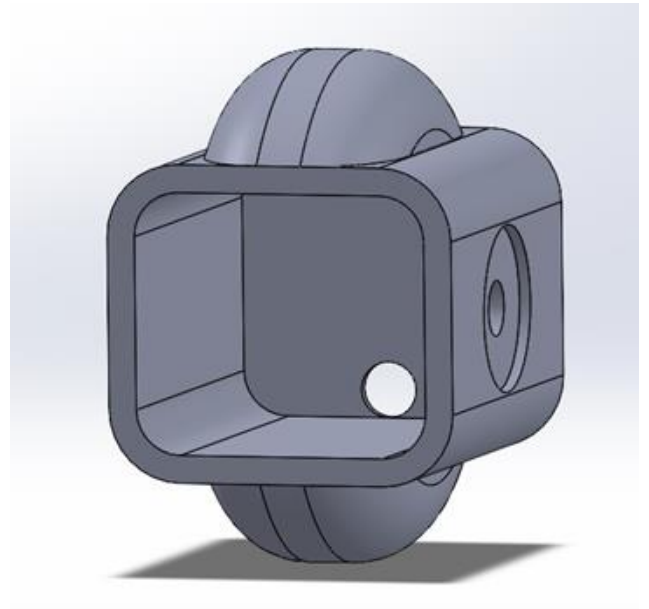
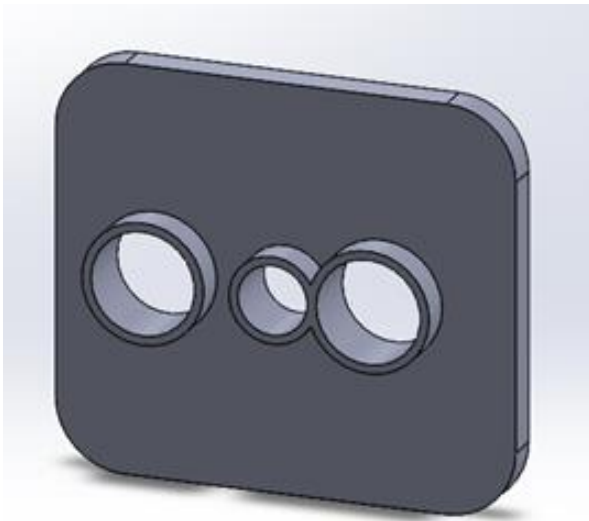


Fig. A3. Third camera-mount design iteration.

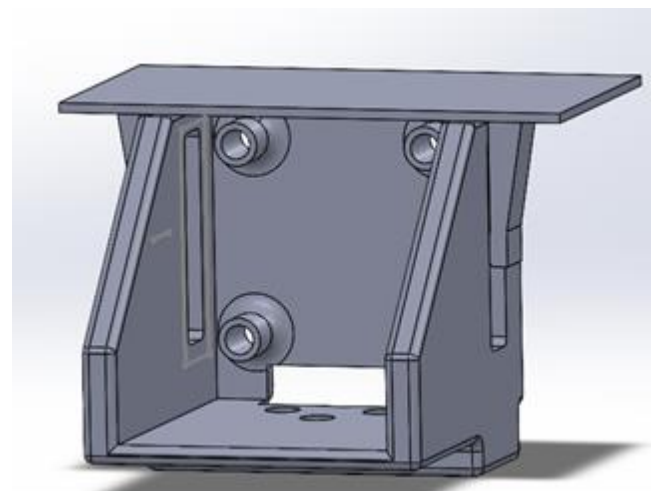
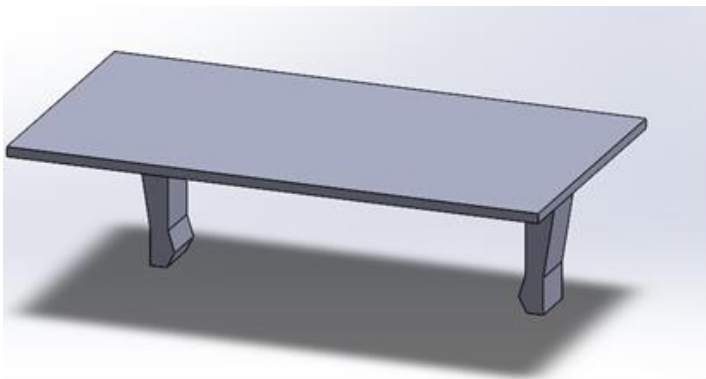
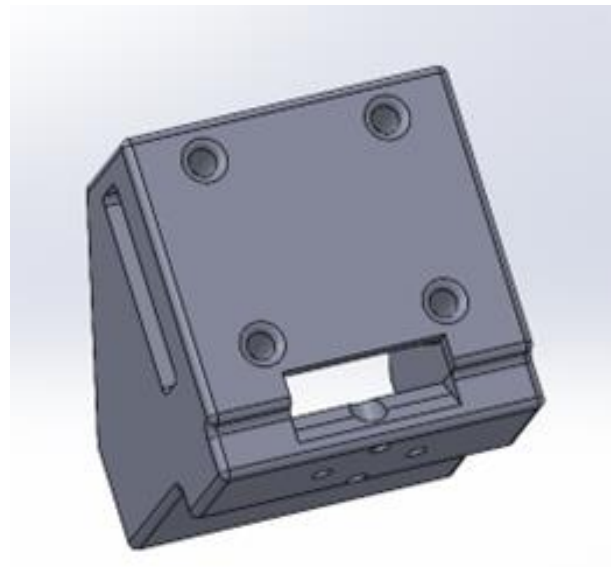
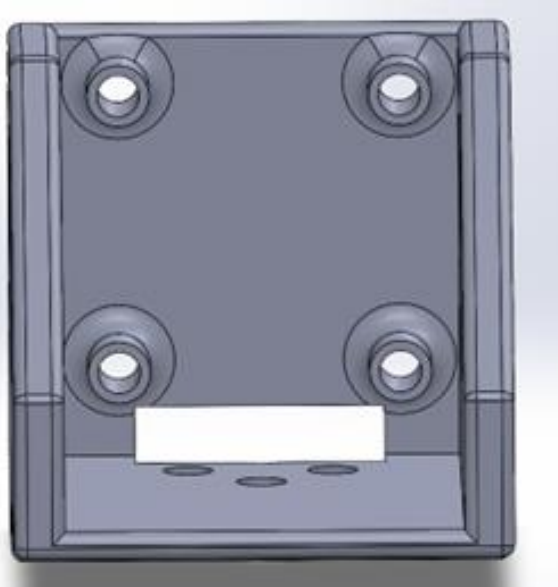
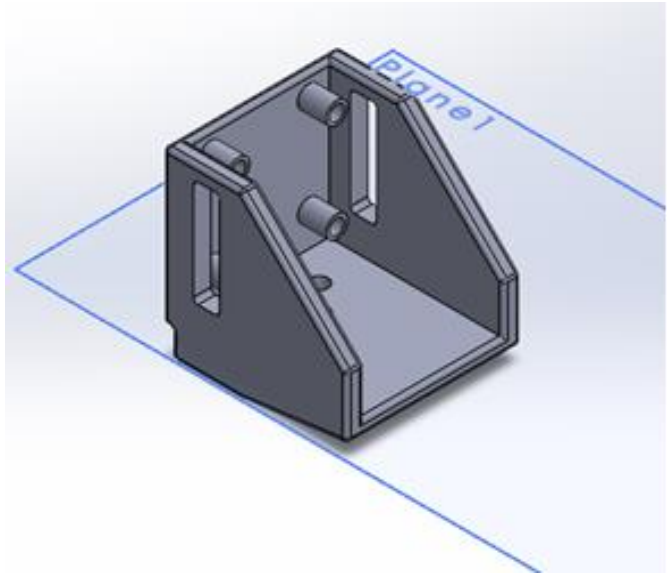
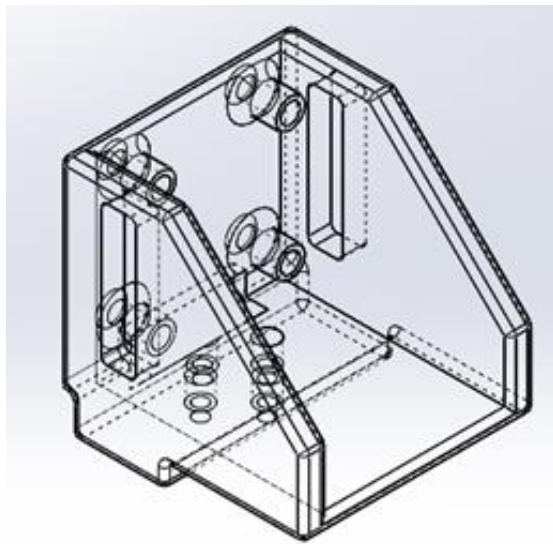
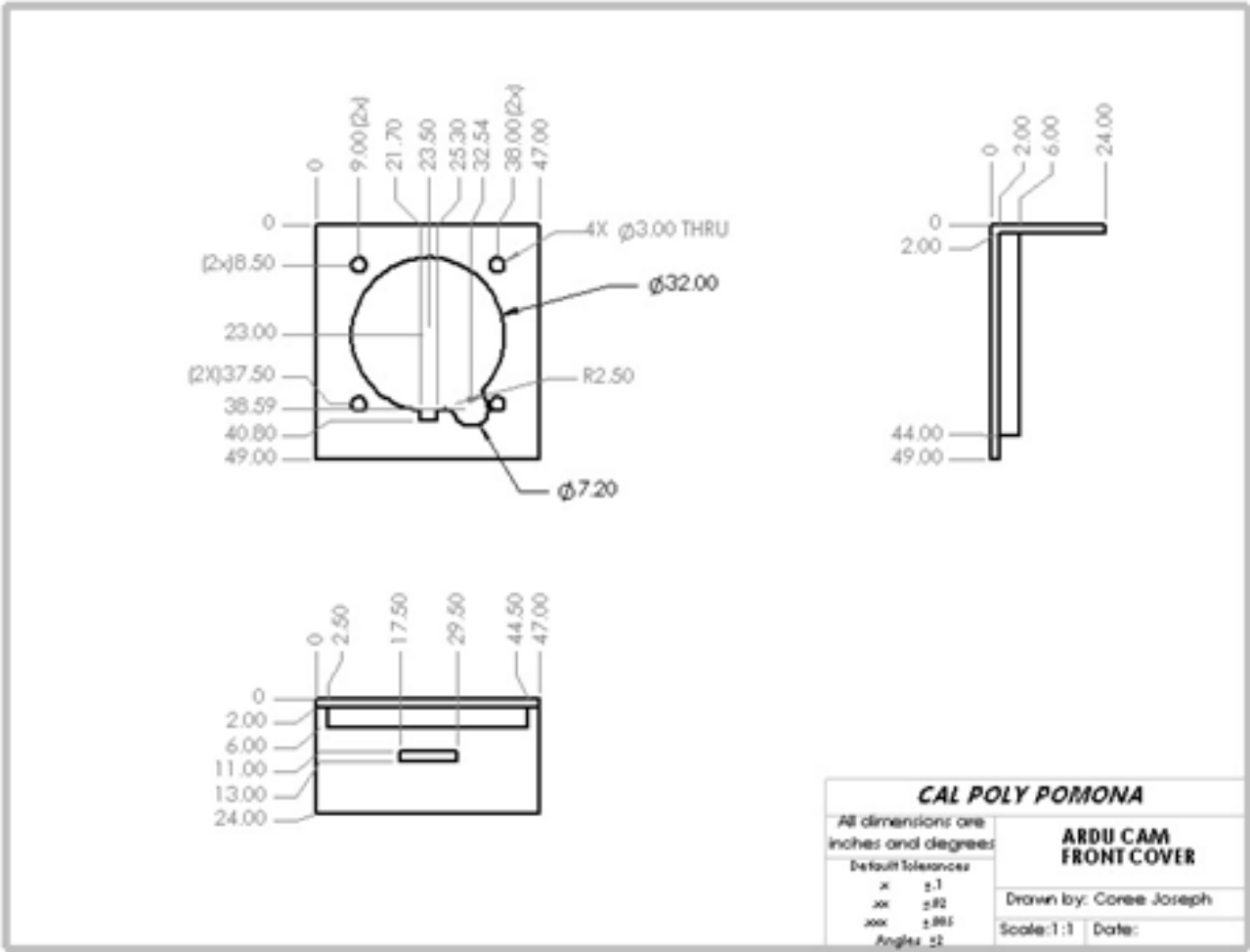
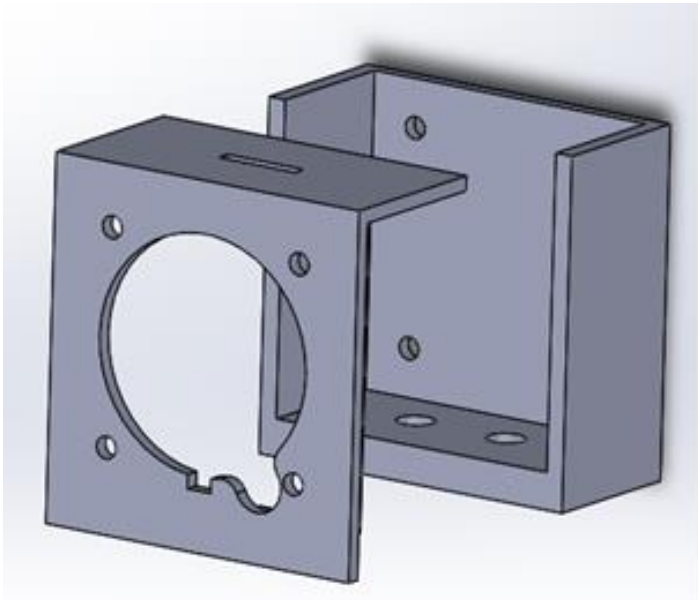
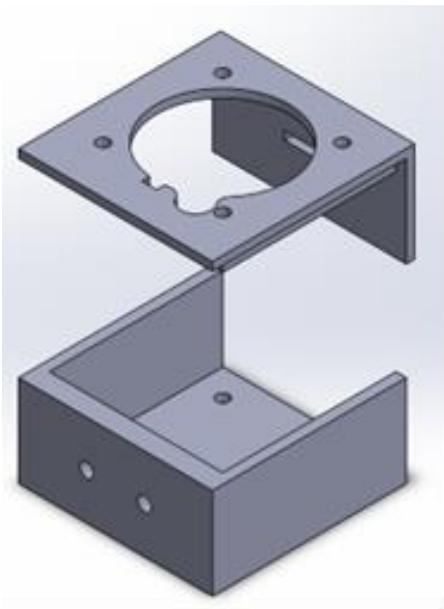
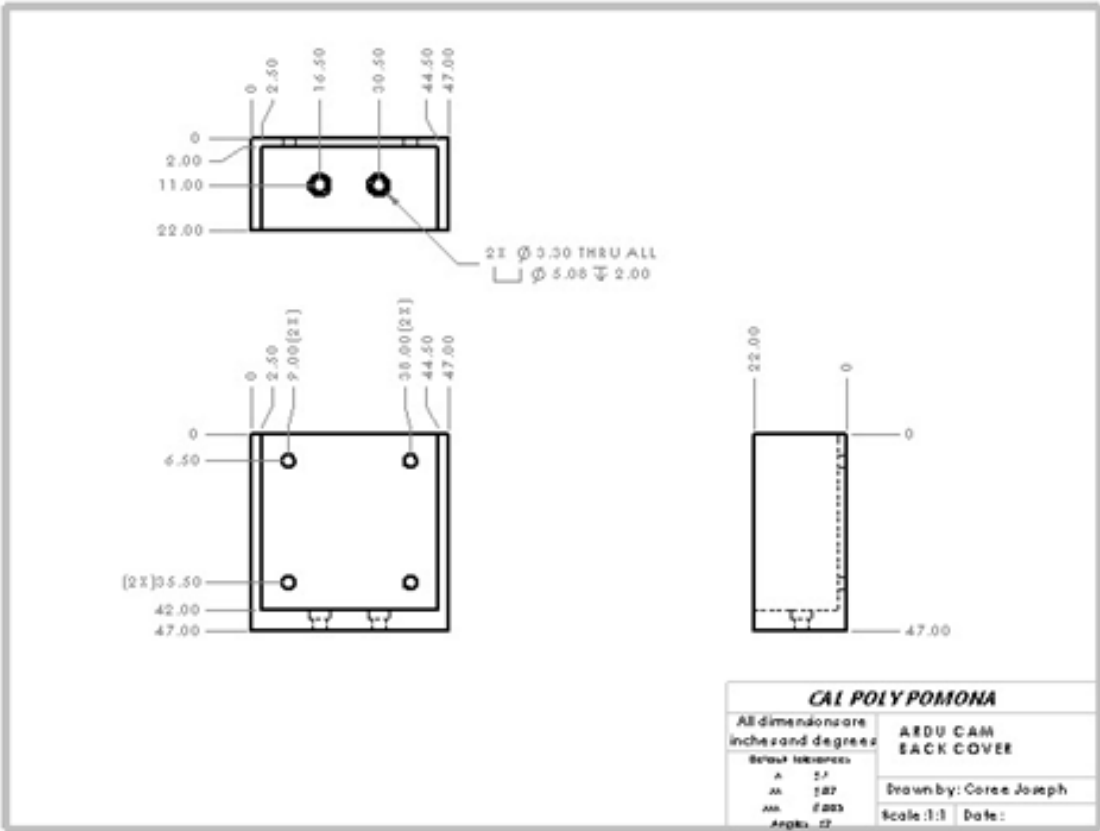
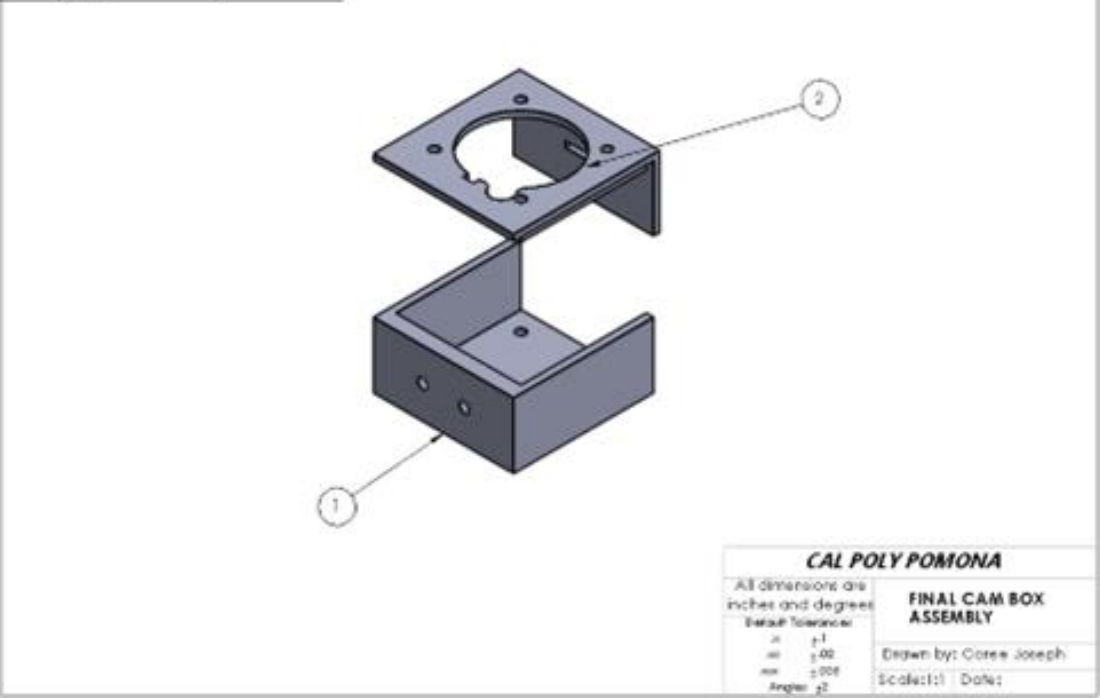


Fig. A4. Final camera-front-cover CAD model and drawing.





| ITEM NO. | DESCRIPTION          | QTY. |
|----------|----------------------|------|
| 1        | Ardu Cam Box Back 2  | 1    |
| 2        | Ardu Cam Box Front 2 | 1    |



**Custom Piece Design Details:**

The only custom-designed part in the system is the camera mount and protective cover used to secure the IMX477 camera to the pan-tilt assembly. The part was developed to provide a simple, lightweight, and easily replaceable mounting solution while also protecting the camera during handling and operation. A PLA-based 3D-printed material was selected because it allowed fast iteration, straightforward fabrication, and easy adjustment of hole placement, clearances, and overall geometry during development. Additive manufacturing was chosen because it was the most practical method for producing and revising the part quickly as the design evolved.

**Integration and Stability Analysis:**

The physical prototype demonstrated successful subsystem integration at a functional level, including camera mounting, pan-tilt operation, processor integration, and basic tracker execution. Current timing analysis focuses on software command-path behavior rather than true physical servo latency, since the system does not include encoder feedback. As a result, current timing results should be interpreted as software dispatch and update behavior only. The prototype is functioning and suitable for bench demonstration, but final quantitative verification still depends on broader structured live testing.

## APPENDIX C – SOFTWARE / PROTOTYPE CODE

```
38 if object_center_x < frame_center_x - threshold:
39     pan_angle -= step_size
40 elif object_center_x > frame_center_x + threshold:
41     pan_angle += step_size
42 pwm.set_pwm(pan_channel, 0, angle_to_pwm(pan_angle))
```

Example control logic in Python for horizontal object tracking using PWM.

### Software Code

The following code is a representative earlier development-stage bench example used for camera bring-up, face-centered motion testing, and PWM command-path verification. It does not represent the final OpenCV DNN-based multi-state tracker described in the main body of this report [2]. It is included as a practical software reference from the development process.

```
import os
import time
import cv2
import mediapipe as mp
from adafruit_servokit import ServoKit

os.environ["XNNPACK_FORCE_SVE"] = "0"
os.environ["GLOG_minloglevel"] = "3"

print("[*] Restarting camera service...")
os.system("sudo systemctl restart nvargus-daemon")
time.sleep(3)

kit = ServoKit(channels=16)

PAN_MIN, PAN_MAX = 60, 120
TILT_MIN, TILT_MAX = 30, 130

pan = 90
tilt = 90
kit.servo[0].angle = pan
kit.servo[1].angle = tilt

def get_pipeline() -> str:
    return (
        "nvarguscamerasrc sensor-id=0 ! "
        "video/x-raw(memory:NVMM), width=1280, height=720, framerate=30/1 ! "
        "nvvidconv ! video/x-raw, width=640, height=480, format=BGRx ! "
        "videoconvert ! video/x-raw, format=BGR ! "
        "appsink drop=True"
    )

cap = cv2.VideoCapture(get_pipeline(), cv2.CAP_GSTREAMER)
```

```

mp_face = mp.solutions.face_detection
face_detection = mp_face.FaceDetection(
    model_selection=0,
    min_detection_confidence=0.5
)

print("[✓] Camera and tracking bench script started.")

if not cap.isOpened():
    print("[!] Camera failed to open. Check CSI connection and Jetson camera service.")
    raise SystemExit(1)

try:
    while True:
        ret, frame = cap.read()
        if not ret:
            break

        rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        results = face_detection.process(rgb)

        if results.detections:
            for detection in results.detections:
                box = detection.location_data.relative_bounding_box

                cx = int((box.xmin + box.width / 2) * 640)
                cy = int((box.ymin + box.height / 2) * 480)

                # Frame-center error
                err_x = 320 - cx
                err_y = 240 - cy

                # Simple deadband + proportional step behavior
                if abs(err_x) > 15:
                    pan += err_x * 0.05
                if abs(err_y) > 15:
                    tilt += err_y * 0.05

                # Clamp motion to software-safe limits
                pan = max(PAN_MIN, min(PAN_MAX, pan))
                tilt = max(TILT_MIN, min(TILT_MAX, tilt))

                kit.servo[0].angle = pan
                kit.servo[1].angle = tilt

                # Visual marker for bench testing
                cv2.circle(frame, (cx, cy), 10, (0, 255, 0), -1)
                break

        cv2.imshow("Jetson Bench Tracking Example", frame)

        if cv2.waitKey(1) & 0xFF == ord("q"):
            break

finally:
    cap.release()
    cv2.destroyAllWindows()

```

# APPENDIX D – COST / VENDOR INFORMATION

| Components                          | Purpose            | Cost            | Store/ Supplier |
|-------------------------------------|--------------------|-----------------|-----------------|
| NVIDIA Jetson Orin Nano Dev Kit     | Main computer      | \$249.00        | Amazon          |
| Arducam IMX477 HQ (auto IR-cut)     | Camera             | \$64.99         | Amazon          |
| Yahboom 2-DoF metal pan-tilt        | Actuation          | \$49.99         | Amazon          |
| PCA9685 PWM driver                  | Servo control      | \$9.99          | Amazon          |
| DC-DC converters (LM2596 + 5 V/5 A) | Power              | \$29.48         | Amazon          |
| IR illuminator (850 nm)             | Night vision light | \$26.99         | Amazon          |
| microSD (64 GB) + NVMe (256 GB)     | Data storage       | \$49.00         | Amazon          |
| Wiring, case, fasteners             | Misc               | \$25.00         | Amazon          |
| <b>TOTAL</b>                        |                    | <b>\$504.44</b> |                 |

## APPENDIX E – FUTURE WORK

Future iterations should include a protective layered skirt or shroud to shield the camera from debris and impacts while maintaining smooth pan-and-tilt movement. Hardware improvements could also include evaluating a higher-resolution Arducam camera and an NVIDIA Jetson Orin Nano Super upgrade to increase available processing bandwidth and improve detection reliability under more demanding conditions. Beyond hardware, the system could be expanded with an approved facial enrollment workflow for verified staff or authorized users. With consent, enrolled individuals could be scanned and stored locally using facial embeddings, allowing BillyBOT to recognize approved users during future interactions without relying on cloud processing. This would build on the current offline tracking architecture while keeping identification data controlled on the device.

Further integration with BillyBOT could allow the camera system to detect when an approved person approaches and trigger a synchronized communication or interaction routine. Additional future features could include gesture-based controls for switching tracking modes, improved target-priority logic, stronger false-positive rejection, and motion-sensor-based wake behavior to reduce power use when no subject is present. These upgrades would require structured coding, validation, and privacy-aware testing to confirm reliable tracking, safe data handling, optimized power distribution, and low system latency.

## REFERENCES

- [1] NVIDIA, “TensorRT Documentation.” Accessed Oct. 28, 2025. Available: <https://docs.nvidia.com/deeplearning/tensorrt/latest/>
- [2] OpenCV, “DNN module documentation.” Accessed Oct. 12, 2025. Available: <https://docs.opencv.org/>
- [3] MIPI Alliance, “CSI-2 Camera Serial Interface.” Accessed Oct. 23, 2025. Available: <https://www.mipi.org/specifications/csi-2>
- [4] NXP Semiconductors, “UM10204: I<sup>2</sup>C-bus specification and user manual, Rev. 6,” 2014. Accessed Oct. 9, 2025. Available: <https://www.nxp.com/docs/en/user-guide/UM10204.pdf>
- [5] Wyze Labs, “Service Plans.” 2024. Accessed Oct. 17, 2025. Available: <https://www.wyze.com/pages/service-plans>
- [6] TechRadar, “Can wireless security cameras work without the Internet?” Aug. 9, 2022. Accessed Oct. 5, 2025. Available: <https://www.techradar.com/news/best-home-security-camera>
- [7] K. Hay, “How criminals are using jammers, deauthers to disrupt Wi-Fi security cameras,” WXYZ Detroit, 2025. Accessed Oct. 30, 2025. Available: <https://www.wxyz.com/news/how-criminals-are-using-jammers-deauthers-to-disrupt-wifi-security-cameras>
- [8] OpenMV, “OpenMV Cam H7 Plus.” 2022. Accessed Oct. 2, 2025. Available: <https://openmv.io/products/openmv-cam-h7-plus>
- [9] I. Carr, “The Hidden Costs of a Server-Based Video System,” CheckVideo, Oct. 17, 2023. Accessed Oct. 26, 2025. Available: <https://www.checkvideo.com/hidden-costs-server-video-system/>
- [10] Arducam, “Day & Night IMX477 HQ Camera for Jetson Orin NX, 12MP Automatic IR-Cut.” 2025. Accessed Oct. 14, 2025. Available: <https://www.arducam.com>
- [11] IEC, “IEC 62676-4: Video surveillance systems for use in security applications — Application guidelines,” 2014. Accessed Oct. 20, 2025.
- [12] ISO, “ISO 9283: Manipulating industrial robots — Performance criteria and related test methods,” 1998. Accessed Oct. 7, 2025.
- [13] Federal Communications Commission (FCC), “47 CFR Part 15, Subpart B — Unintentional radiators.” Accessed Oct. 11, 2025. Available: <https://www.ecfr.gov/>

- [14] IEC, “IEC 61000-4-5: Electromagnetic compatibility (EMC) — Surge immunity test,” 2014. Overview accessed Oct. 4, 2025 via EMC FastPass AN4275.
- [15] IEEE Standards Association, “IEEE 1857-2021 — Standard for Real-Time 3D Visual Computing,” 2021. Accessed Oct. 19, 2025.