Autonomous Security System

Diego Iván Morales Gallardo

February 16, 2025

Contents

1 Description of the Problem to Be Developed

In modern security systems, traditional methods often fall short when it comes to monitoring vast open areas. Limited resources, delayed responses, and the constraints of continuous human observation can compromise overall safety. Our project addresses these challenges by simulating a coordinated multi-agent security system that integrates advanced computer vision and autonomous response units.

1.1 Expanded Context and Objectives

The simulation is designed to overcome typical shortcomings by:

- Rapidly detecting unusual activities through intelligent cameras using YOLOv5.
- Promptly dispatching drones to verify alerts and stream live footage for further evaluation.
- Enabling security personnel to collaborate in a structured decision-making process, ensuring that each alert is properly validated.

This hybrid approach ensures that threats are not only detected swiftly but also confirmed with high accuracy, balancing automated responses with critical human oversight.

1.2 Operational Workflow

The system follows a streamlined process:

1. Surveillance cameras continuously monitor the environment and analyze images using YOLOv5.
2. When a potential threat is identified, an alert is issued via a structured message protocol (e.g., KQML), prompting the deployment of a drone.
3. The drone navigates to the location of the alert, provides live video feed, and gathers additional data.
4. Security personnel review the incoming data, utilizing a collaborative voting mechanism if necessary, to decide whether to escalate or dismiss the alert.
5. Upon resolution, the drone safely returns to its base, ready for subsequent tasks.

2 Agents and Their Roles

The project involves multiple agents, each with specific roles and responsibilities within the simulation. These agents work collaboratively to detect, verify, and respond to potential security threats. The key agents and their roles are as follows:

2.1 Surveillance Cameras (Camera Agents)

Role: Monitor the environment and detect potential threats using computer vision.

Responsibilities:

- Capture images of the surroundings at regular intervals.
- Process images through the YOLOv5 object detection model to identify objects of interest, such as intruders (e.g., a person).
- Send alerts to other agents, such as drones or the simulation server, when a threat is detected.
- Log system activity and detection results for further analysis.

2.2 Drone (Drone Agent)

Role: Respond to alerts from surveillance cameras and confirm potential threats.

Responsibilities:

- Take off and patrol designated areas autonomously while maintaining optimal battery usage.
- Receive alerts from cameras and navigate to the location of potential threats.
- Use an onboard camera to capture and process images for threat verification.
- Respond to environmental factors like wind and battery constraints while maintaining operational efficiency.
- Report back to the simulation server or security guard with findings and recommendations.
- Return to the landing station when tasks are complete or battery levels are low.

2.3 Security Guard (Guard Agent)

Role: Provide human-like oversight and final decision-making for the system.

Responsibilities:

- Analyze drone findings and determine if detected threats are genuine or false alarms.
- Coordinate decision-making processes by initiating a simulated voting mechanism to evaluate the severity of threats.
- Maintain logs and oversee the system's operations to ensure optimal performance.

2.4 Robber (Robber Agent)

Role: Simulate the presence of an intruder within the environment.

Responsibilities:

- Provide realistic scenarios for cameras and drones to detect and respond to.

2.5 Simulation Server

Role: Act as the central controller and data repository for the simulation.

Responsibilities:

- Manage communication between agents, including sending and receiving alerts, status updates, and commands.
- Log all interactions, metrics, and reports from agents.
- Serve as the platform for analyzing the overall performance of the system.

2.6 YOLOv5 Model

Role: Provide object detection capabilities for cameras and drones.

Responsibilities:

- Process images sent by the surveillance cameras and drone to identify objects of interest.
- Return detection results, including identified objects, confidence levels, and coordinates, for further decision-making.

2.7 Environment and Extended Agent Functions

In addition to the primary agent roles, the simulation environment is modeled as a realistic open area with designated patrol zones and a dedicated drone landing station. Environmental factors—such as wind—are simulated to challenge and assess the adaptability of drone operations. Moreover, the system designates additional responsibilities:

- **Security Personnel** actively evaluate alerts and orchestrate coordinated responses, issuing commands to both drones and cameras.
- **Surveillance Cameras** act as static observers, continuously monitoring their assigned areas and relaying precise location data of detected anomalies.
- **Drones** serve as agile units that not only patrol the area but also verify potential threats by sharing real-time updates with other agents.

3 Relationships Between Agents

The simulation relies on a network of interactions between agents to replicate the dynamics of a real-world security system. Each agent performs specialized tasks and communicates with others to detect, verify, and respond to potential security threats. Below is an overview of the relationships between agents:

3.1 Surveillance Cameras and Simulation Server

Relationship: Surveillance cameras act as monitoring devices, continuously capturing images of the environment and sending them to the simulation server for processing.

Key Interactions:

- The camera sends detection results, including detected objects and their locations, to the server.
- The server logs the data, evaluates the situation, and forwards alerts to relevant agents (e.g., the drone or guard).

3.2 Surveillance Cameras and Drone

Relationship: Cameras alert the drone when a potential threat is detected.

Key Interactions:

- Upon detecting a suspicious object (e.g., a person), a camera sends the drone a Call For Proposal (CFP) via the Contract Net Protocol (CNP) to investigate the identified location.
- The drone evaluates its current status (e.g., battery level, position) and sends a proposal back to the camera, accepting or rejecting the task.

3.3 Drone and Simulation Server

Relationship: The drone serves as a mobile verification unit, updating the simulation server on its actions and findings.

Key Interactions:

- The drone sends periodic updates to the server, including battery status, position, and detection results.
- The server logs the drone's activities and coordinates communication between the drone and other agents.

3.4 Drone and Security Guard

Relationship: The drone relies on the security guard for final decisions regarding detected threats.

Key Interactions:

- After capturing evidence, the drone notifies the guard of its findings and awaits further instructions.
- The guard evaluates the data, conducts a voting process (if necessary), and provides directives to the

drone (e.g., return to base, or dismiss the alert).

3.5 Security Guard and Simulation Server

Relationship: The security guard oversees the simulation and acts as a decision-maker, coordinating with the server for data analysis and action logging.

Key Interactions:

- The guard logs decisions and receives information about alerts and the status of other agents from the server.
- The server serves as a communication hub, ensuring the guard has the necessary data to make informed decisions.

3.6 Robber, Surveillance Cameras and Drone

Relationship: The robber serves as a test subject for the surveillance cameras and drone, simulating an intruder.

3.7 YOLOv5 Model, Surveillance Cameras and Drone

Relationship: YOLOv5 acts as the detection engine, assisting cameras and the drone in identifying potential threats.

Key Interactions:

- Cameras and drones send image data to the YOLOv5 server for processing.
- YOLOv5 returns detection results, including detected objects, confidence levels, and coordinates, enabling further decision-making by the respective agents.

3.8 Robber and Security Guard

Relationship: The robber indirectly interacts with the security guard by influencing decisions.

Key Interactions:

- The guard analyzes the drone's findings regarding the robber and decides whether to escalate or dismiss the threat.

3.9 Communication and Coordination Mechanisms

Robust communication is vital for system performance. Agents exchange information using standardized protocols (such as KQML), ensuring that alerts initiated by cameras prompt timely responses from drones. Security personnel further enhance this network by engaging in a collective review process to validate each alert.

3.9.1 Auction-Based Task Allocation

Task assignments are dynamically managed through a Contract Net Protocol (CNET), wherein agents (drones and cameras) submit proposals based on factors like proximity and resource status. This auction-based system ensures that the most capable agent is tasked with responding to a given alert.

3.9.2 Collaborative Threat Assessment

To minimize false alarms, a voting mechanism is activated among security personnel when ambiguity arises. This collaborative process ensures that decisions to escalate or dismiss an alert are both well-informed and democratically validated.

4 Agent Properties with Justification for Each Property

4.1 Surveillance Cameras (Camera Agents)

Properties:

- **surveillanceCamera (Camera):** References the camera component of the object. This property is essential for capturing images of the simulation environment.
- **cameraId (String):** A unique identifier for each camera. This allows the simulation server and other agents to identify the source of detection data.
- **serverUrl (String):** Specifies the URL for the YOLOv5 server. Enables communication with the detection model for image processing.
- **captureInterval (Float):** Defines the time interval (in seconds) between image captures. Helps balance detection frequency and computational resource usage.
- **captureWidth and captureHeight (Integers):** Specify the resolution of the captured images. Higher resolution improves detection accuracy but requires more processing power.
- **thiefDetected (Static Boolean):** Tracks whether a thief has been detected by any camera. Shared across all camera agents to coordinate their behavior.
- **totalImagesSent (Integer):** Counts the total number of images sent to the YOLOv5 server. Helps measure performance and activity levels.
- **successfulDetections (Integer):** Tracks the number of successful detections. Used for evaluating the system's effectiveness.

- **totalLatency and latencySamples (Floats and Integers):** Measure the time taken for the YOLOv5 server to process images and respond. Crucial for analyzing system responsiveness.

Justification: These properties ensure that each camera agent can monitor the environment effectively, detect threats, and communicate findings to the rest of the system.

4.2 Drone (Drone Agent)

Properties:

- **landingStationPosition (Vector3):** Specifies the position of the drone's base. Used for returning and landing operations.
- **cameraDetectionTime (Float):** Stores the time at which a camera detected a thief. Helps calculate response times.
- **currentBatteryLevel (Float):** Tracks the drone's battery level. Enables decisions such as returning to base when the battery is low.
- **takeOffHeight and moveSpeed (Floats):** Define the altitude for patrolling and the movement speed of the drone. These parameters control the drone's navigation.
- **patrolPoints (List of Vector3):** Contains predefined waypoints for patrolling. Enables the drone to autonomously cover specific areas.
- **windStrength (Float):** Represents the effect of wind on the drone's movement. Adds realism to the simulation by simulating environmental challenges.
- **thiefDetected (Boolean):** Tracks whether the drone has detected a thief. Coordinates with cameras and the guard to confirm threats.
- **detectionConfirmationTimes (List of Floats):** Stores the times taken to confirm detections. Useful for evaluating system performance.
- **droneTotalLatency and droneLatencySamples (Floats and Integers):** Measure the drone's communication efficiency with the YOLOv5 server.

Justification: These properties enable the drone to operate autonomously, perform threat verification, and adapt to environmental and operational constraints.

4.3 Security Guard (Guard Agent)

Properties:

- **status (String):** Represents the guard's operational state (e.g., "Idle", "Analyzing"). Tracks the agent's role in decision-making processes.
- **messageQueue (Simulated Messages):** Used for receiving and processing alerts or votes. Coordinates with drones and cameras.

- **voteResults (Boolean Decision):** Simulates the result of a voting process to assess threats. Adds an element of human-like decision-making to the system.

Justification: These properties allow the guard to act as a human decision-maker, ensuring threats are analyzed thoroughly and false alarms are minimized.

4.4 Robber (Robber Agent)

Properties:

- **position (Vector3):** Specifies the current location of the robber. Allows cameras and drones to locate and respond to the intruder.
- **status (String):** Tracks the robber's activity (e.g., "Active", "Idle"). Adds variability to the simulation by simulating different behaviors.
- **simulationEffectiveness (Implicit):** The robber's presence indirectly measures the system's ability to detect and respond to threats.

Justification: These properties make the robber a dynamic element in the simulation, providing realistic scenarios to test the security system's capabilities.

4.5 Simulation Server

Properties:

- **agentRegistry (Dictionary):** Tracks all agents, including their properties and states. Centralizes management and coordination.
- **logSystem (List of Strings):** Records interactions and decisions. Enables debugging and performance analysis.
- **performanceMetrics (Dictionaries):** Captures key metrics like detection rates, latency, and battery usage. Supports system evaluation and optimization.
- **messageQueue (Queue):** Manages inter-agent communication. Facilitates smooth and asynchronous operations within the simulation.

Justification: These properties allow the simulation server to act as the central communication hub, ensuring seamless integration and comprehensive data analysis.

4.6 YOLOv5 Model

Properties:

- **model (Pretrained YOLOv5):** The detection model used for identifying objects in images. Essential for accurate threat detection.
- **confidenceThreshold (Float):** The minimum confidence required for a detection to be considered valid. Balances precision and recall.
- **objectClasses (List of Strings):** Specifies the classes of objects the model can detect (e.g., ”person”). Aligns with the system’s objectives.

Justification: These properties define the YOLOv5 model’s capabilities, ensuring it provides accurate and reliable object detection for cameras and drones.

5 Agent Class Diagrams

This section presents the class diagrams for each agent in the simulation. These diagrams illustrate the structure, attributes, and relationships between different agents in the system.

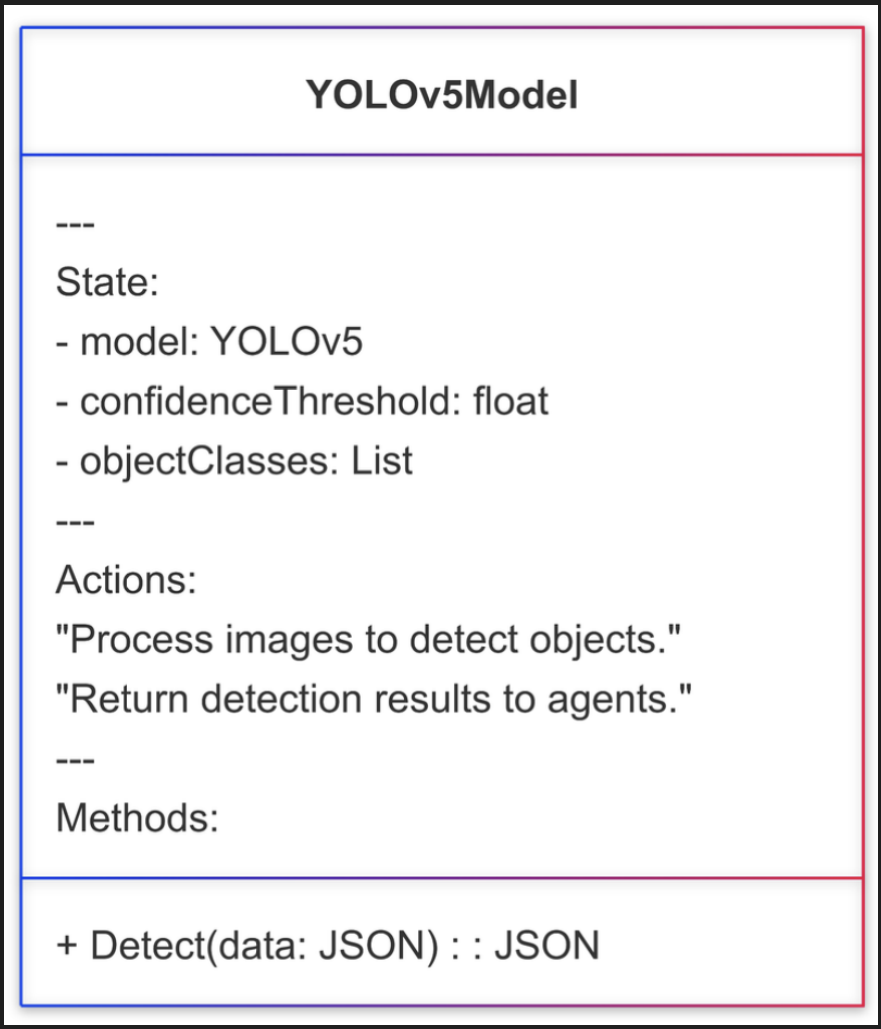

Figure 1: Class Diagram of YOLOv5 Model

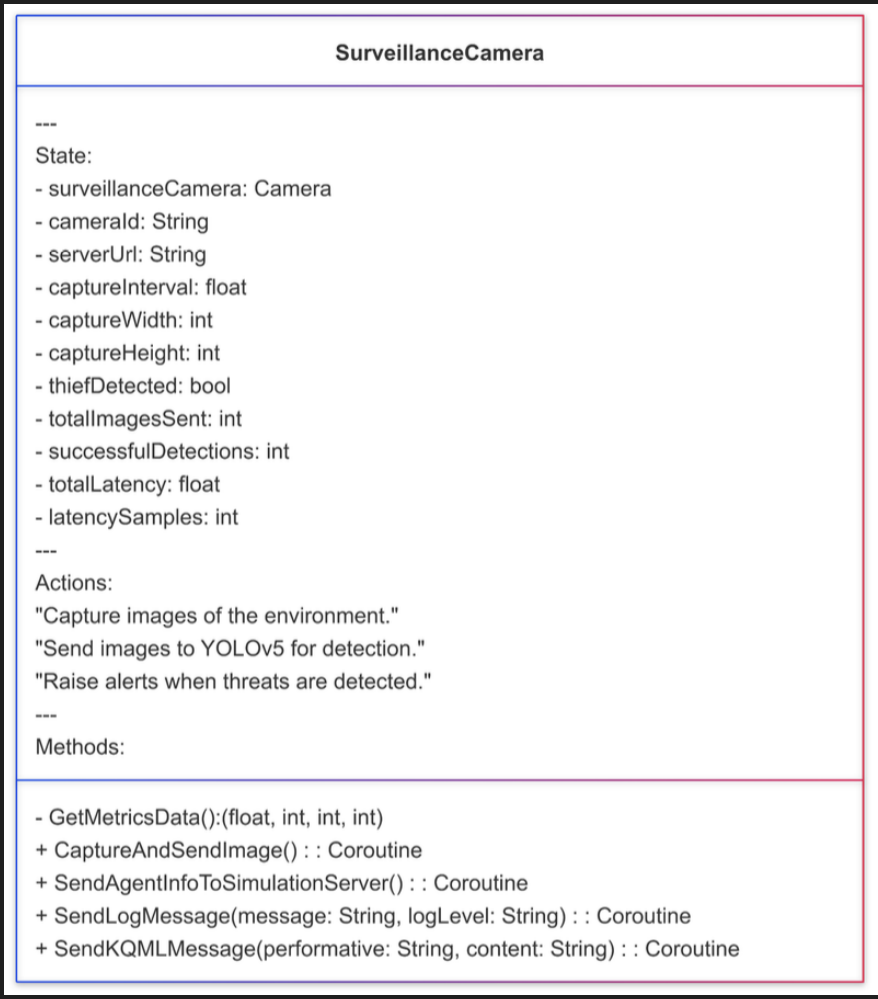

Figure 2: Class Diagram of Surveillance Camera Agent

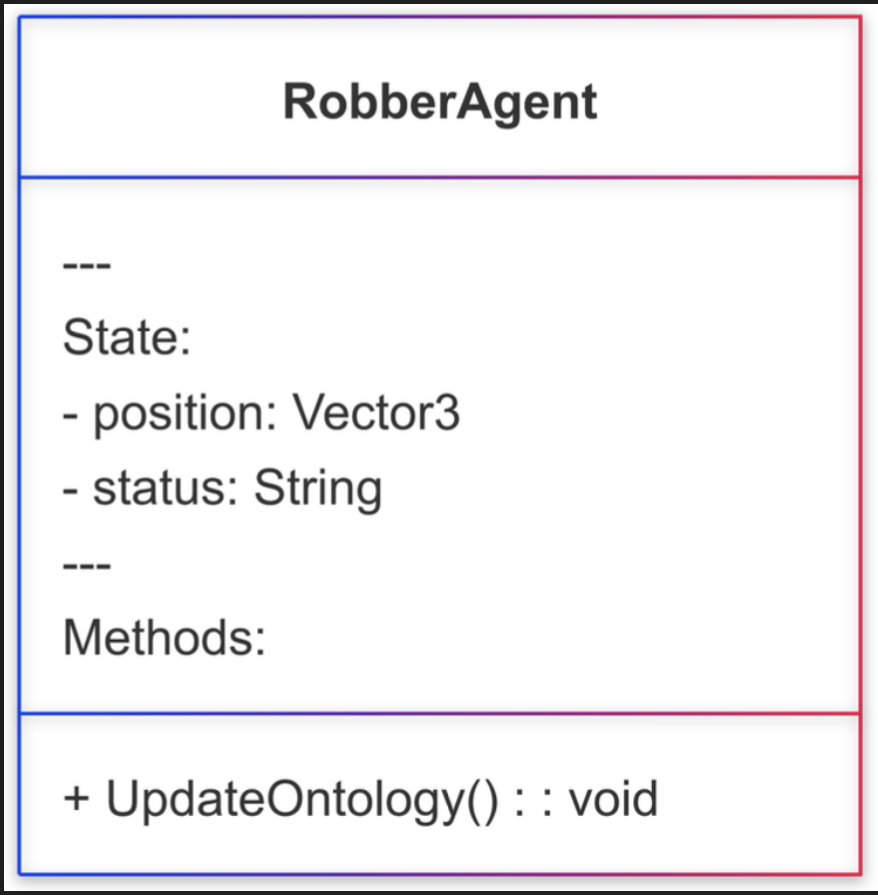

Figure 3: Class Diagram of Robber Agent

Figure 4: Class Diagram of Security Guard Agent

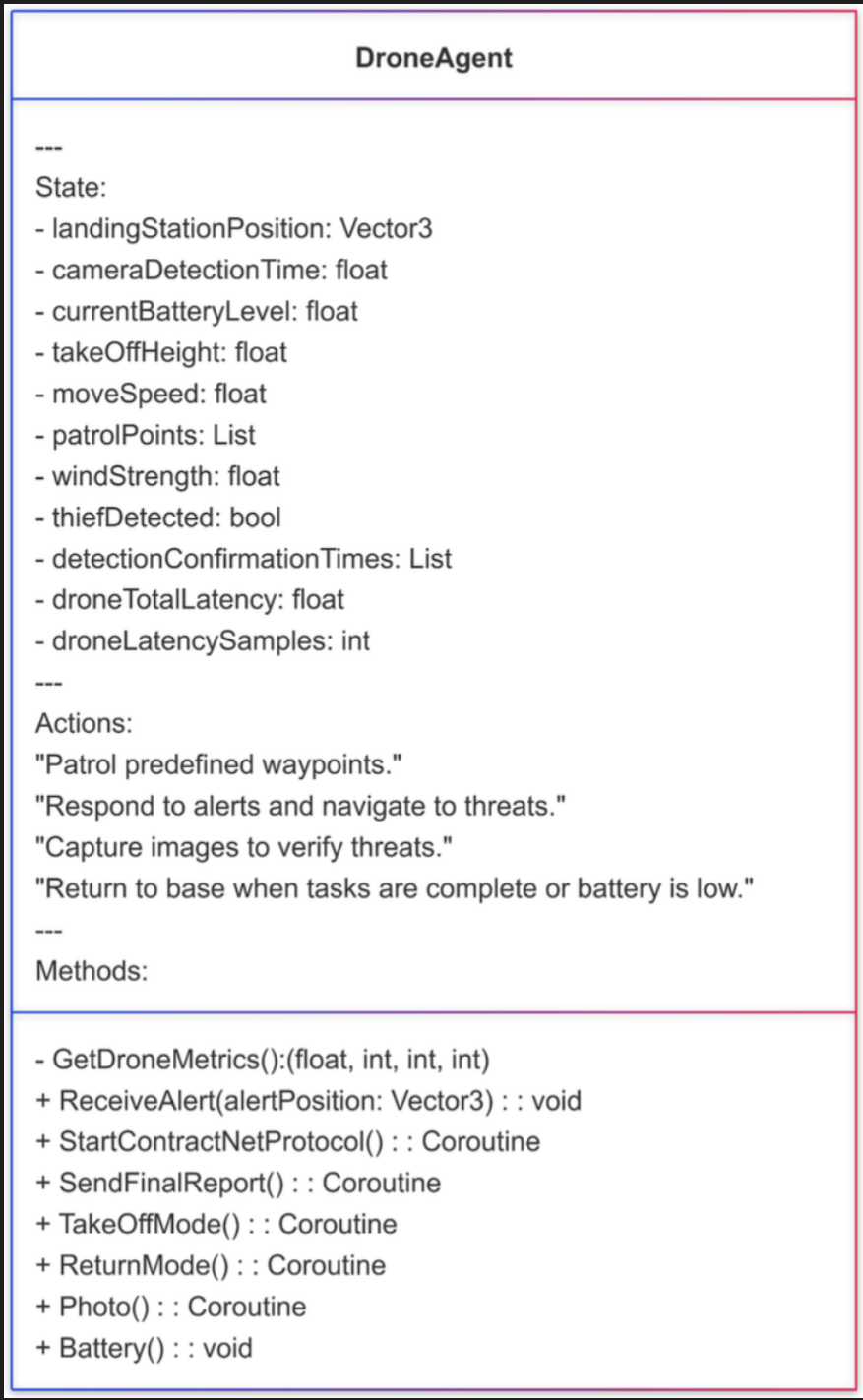

Figure 5: Class Diagram of Drone Agent

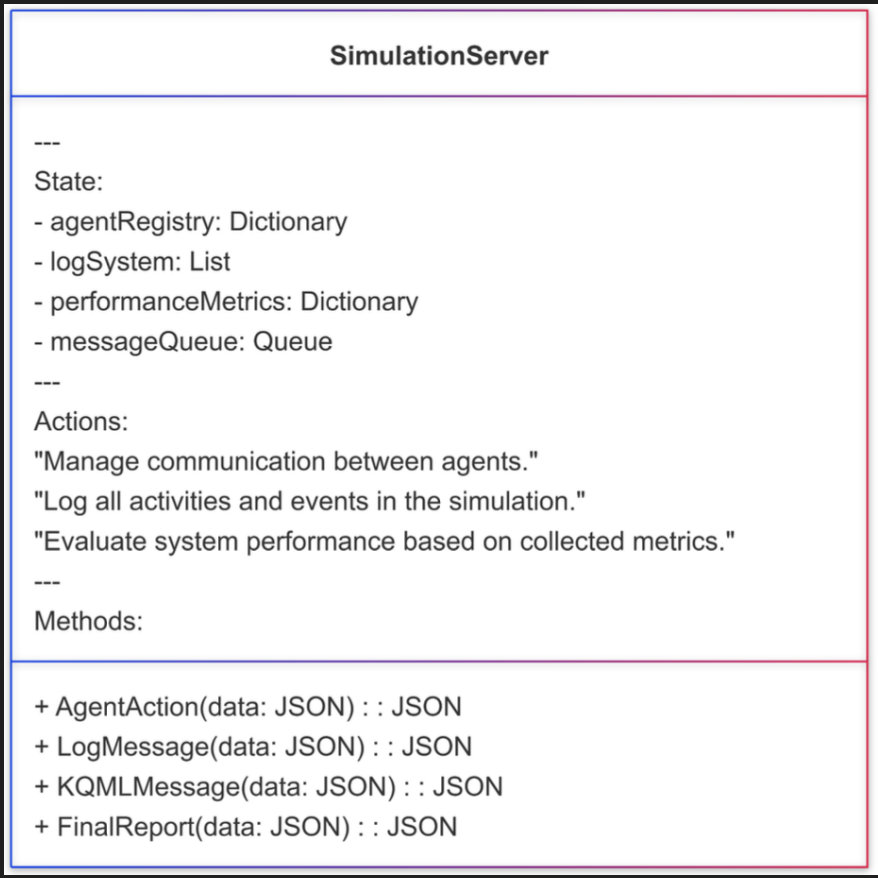

Figure 6: Class Diagram of Simulation Server

6 Metrics and Success Measurement of the Simulation

The success of the simulation is assessed through quantitative metrics derived from agent interactions, system performance, and overall outcomes. Below is an outline of key metrics, their interpretations, and calculations.

6.1 Key Metrics

6.1.1 Detection Accuracy

- **Definition:** The percentage of successful detections (true positives) relative to the total images processed by cameras and drones.
- **Calculation:**

$$\text{Detection Accuracy (\%)} = \left(\frac{\text{Successful Detections}}{\text{Total Images Processed}} \times 100 \right) \tag{6.1}$$

6.1.2 Average Latency

- **Definition:** The average time taken by the YOLOv5 model to process images and return results.
- **Calculation:**

$$\text{Average Latency (s)} = \frac{\text{Total Latency}}{\text{Latency Samples}} \tag{6.2}$$

6.1.3 Battery Efficiency

- **Definition:** The average battery consumption per patrol cycle for the drone.
- **Calculation:**

$$\text{Battery Consumption per Cycle (\%)} = \frac{\text{Total Battery Consumed}}{\text{Completed Patrol Cycles}} \tag{6.3}$$

6.1.4 Time-to-Confirmation

- **Definition:** The time between the initial detection of a threat by cameras and confirmation by the drone or security guard.
- **Interpretation:** Lower values indicate efficient coordination among agents.

6.1.5 Distance Traveled by Drone

- **Definition:** The total distance traveled by the drone during the simulation.
- **Calculation:** Sum of distances moved between consecutive positions.

6.1.6 False Alarms

- **Definition:** The number of detections classified as threats that were later dismissed as false alarms by the security guard.
- **Interpretation:** Lower false alarm rates indicate better system reliability.

6.2 Enhanced Performance Metrics and Evaluation Criteria

Beyond the conventional performance metrics, our evaluation framework also tracks:

- **Patrol Time:** The effectiveness with which the drone covers its designated area.
- **Battery Usage:** Detailed monitoring of energy consumption per patrol cycle.
- **Confirmation Time:** The interval between the initial detection of a potential threat and its final validation.

These metrics provide a comprehensive picture of both operational efficiency and resource management.

6.3 Success Criteria

The simulation is considered successful if:

- A suspicious individual is detected and confirmed within the operational cycle.
- The drone retains sufficient battery levels to safely return to its base.
- Security personnel accurately discern between genuine threats and false alarms, ensuring that appropriate responses are triggered.

7 System Architecture

7.1 Deductive Architecture Approach

The system is built on a deductive framework where predefined rules govern decision-making. Autonomous drones act on real-time alerts by considering available resources and environmental conditions, while critical decisions are reviewed by human operators. This hybrid model blends the rapid response of automated systems with the nuanced judgment of human oversight, resulting in a robust and reliable security solution.

8 Drone Performance: Battery vs Distance Over Time

The performance of the drone is evaluated by analyzing its battery consumption in relation to the distance it travels over time. This metric is crucial for assessing the efficiency of the drone in patrolling and responding to threats while managing power consumption.

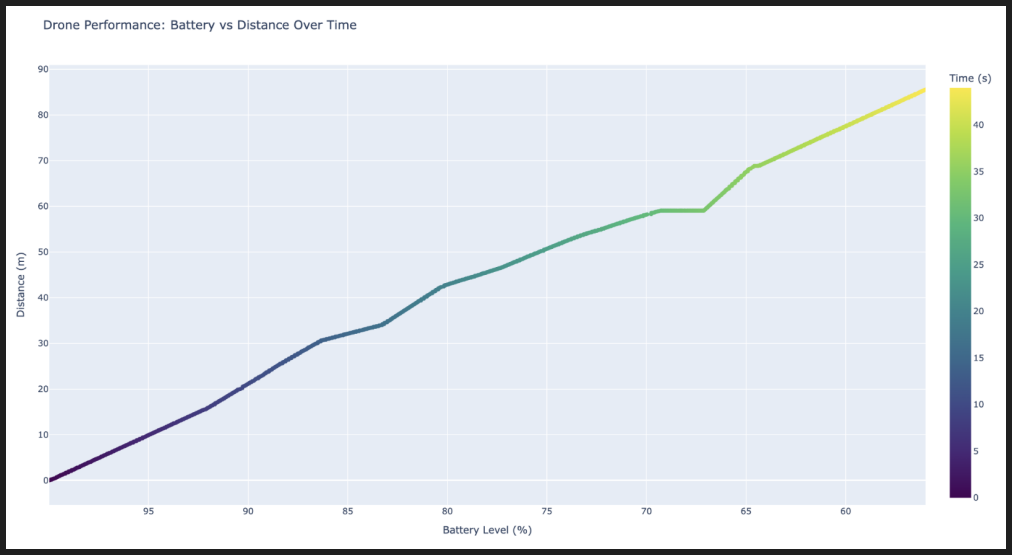

Figure 7: Drone Performance: Battery vs Distance Over Time

The graph illustrates the correlation between the drone's battery depletion and the distance covered. A well-optimized system ensures that the drone maximizes its patrol coverage while maintaining sufficient battery levels to return to its base.

9 Sequence Diagrams of Interaction Protocols

This section presents the sequence diagrams illustrating the interactions between agents in different scenarios of the simulation. These diagrams help visualize the communication and coordination between the surveillance cameras, drones, security guards, and the simulation server.

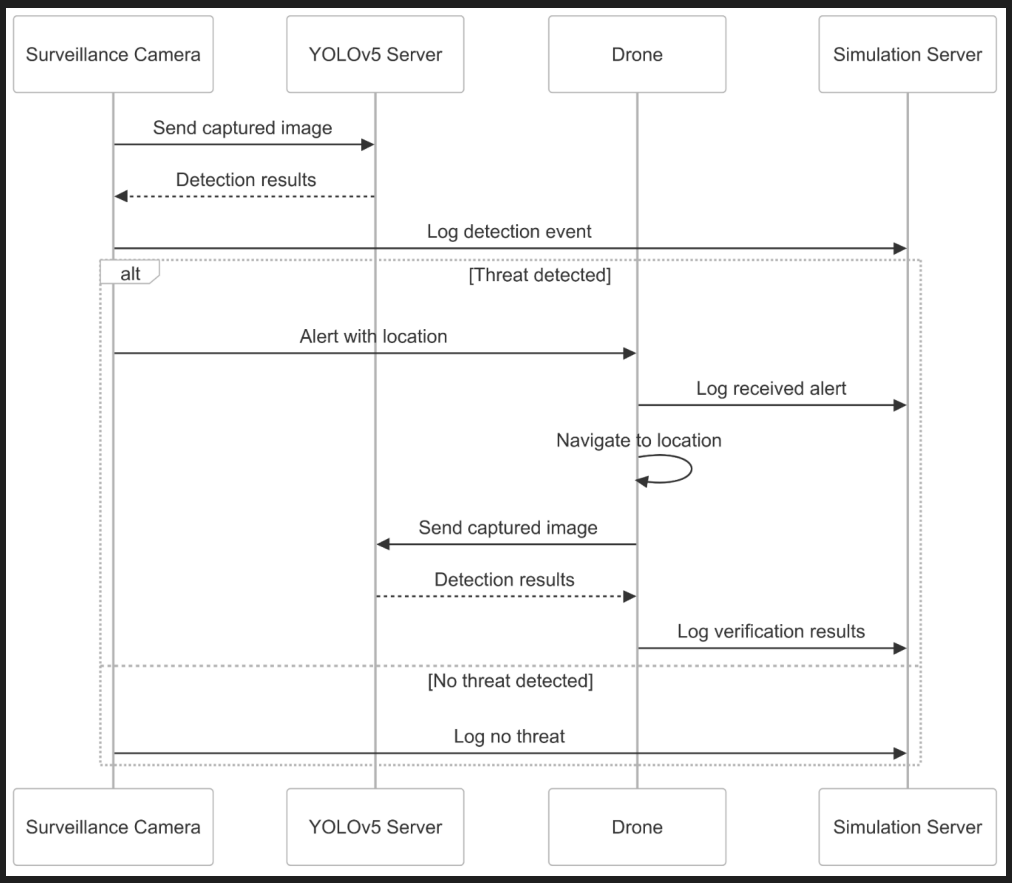

Figure 8: Sequence Diagram: Detection and Alert Protocol

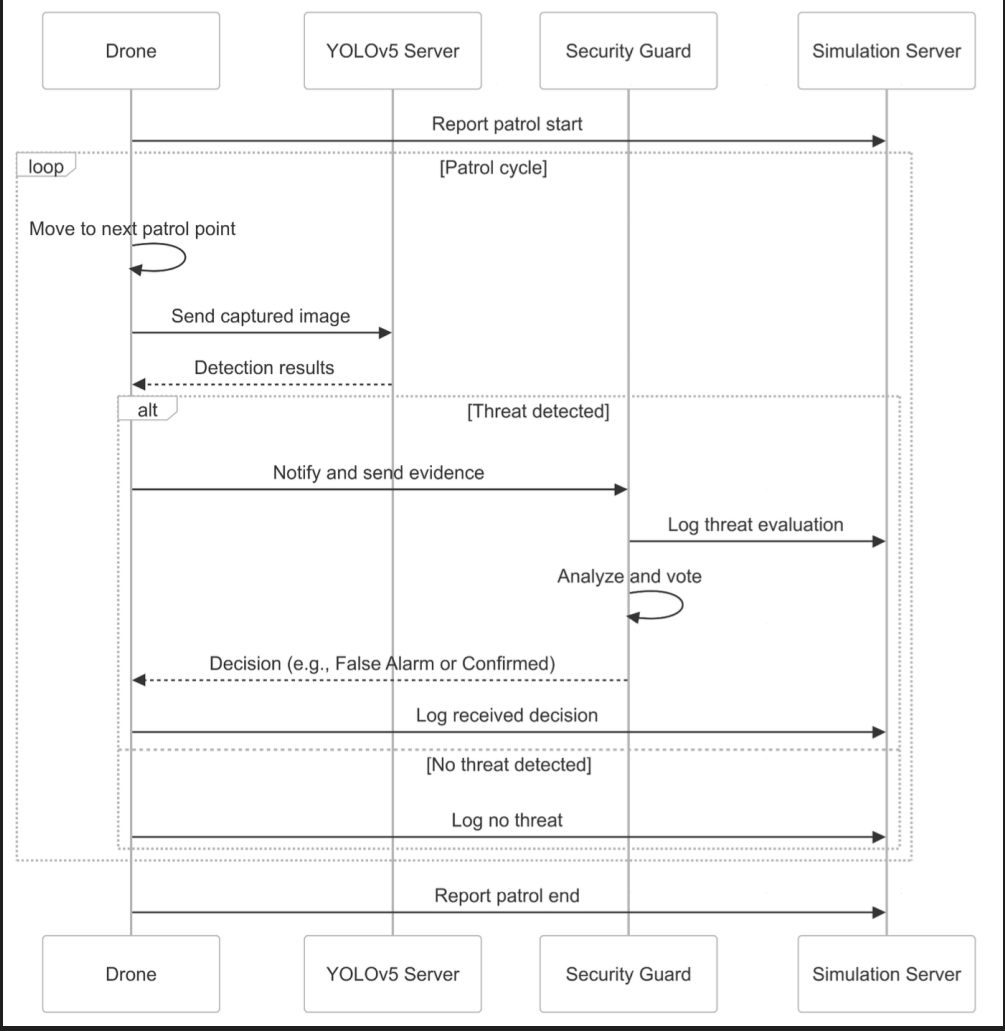

Figure 9: Sequence Diagram: Drone Patrol and Threat Verification

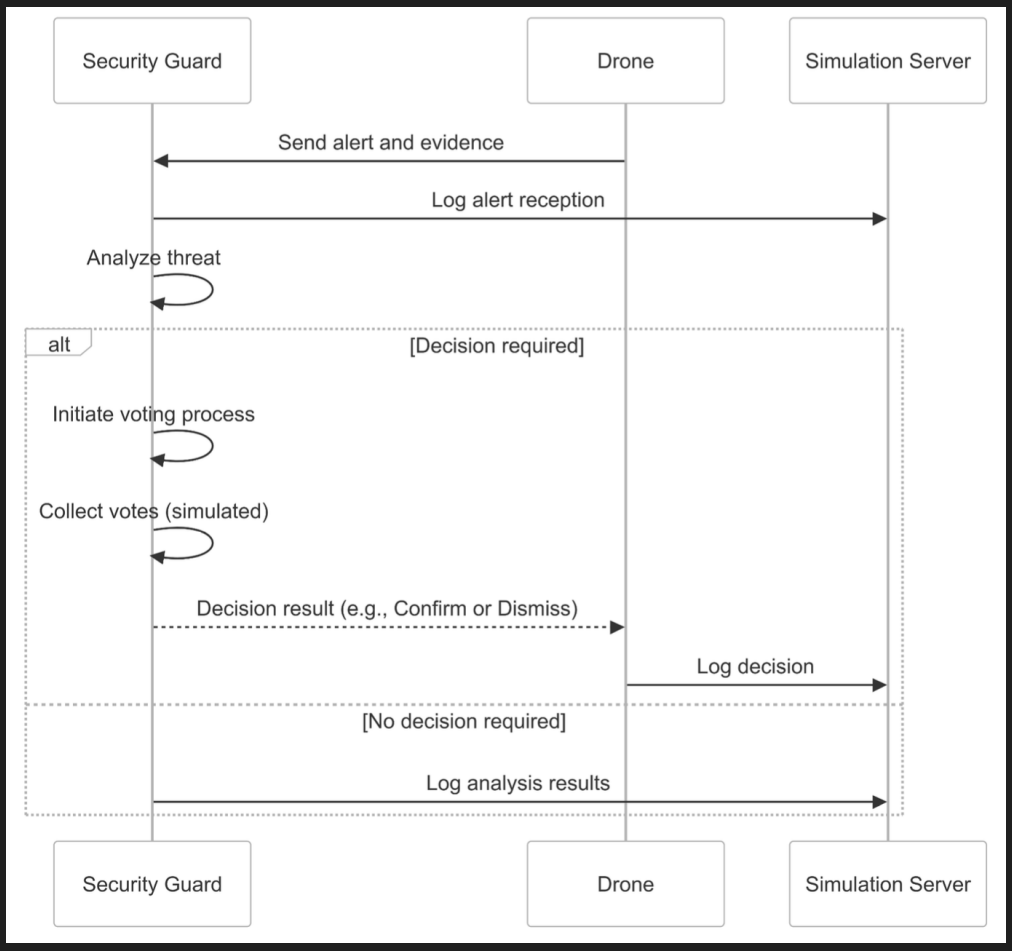

Figure 10: Sequence Diagram: Security Guard Decision-Making Protocol

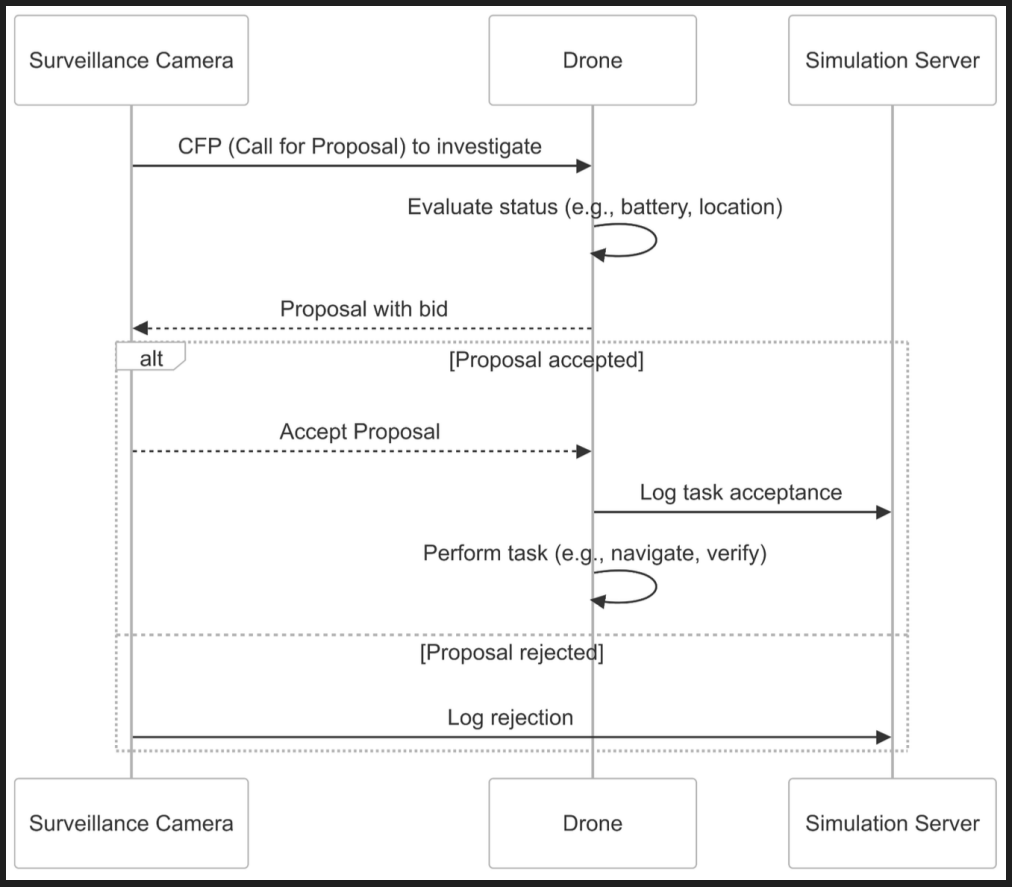

Figure 11: Sequence Diagram: Contract Net Protocol for Task Assignment

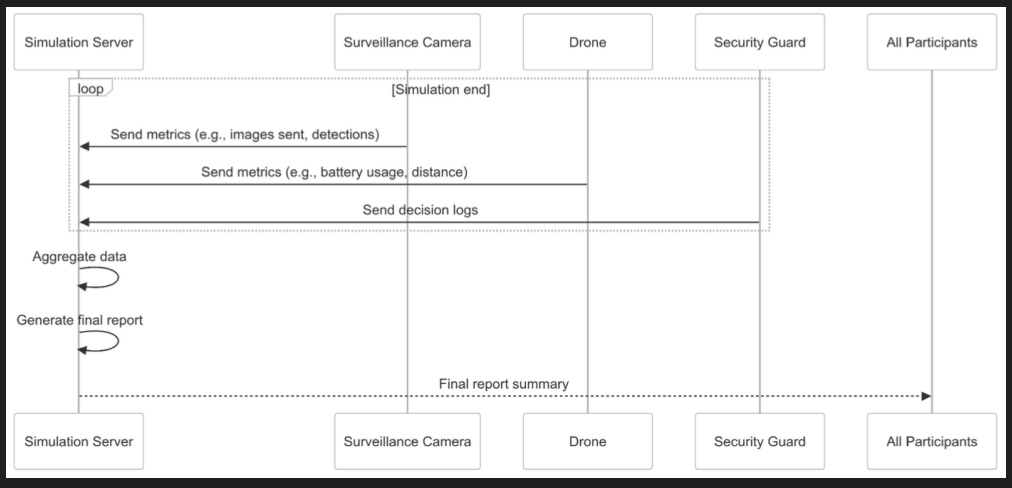

Figure 12: Sequence Diagram: Final Report Generation