Technical Architecture and Implementation Strategy for 'Agri Twin': An Autonomous Edge-AI Crop Scanning System

1. Introduction: The Convergence of Precision Robotics and Digital Twin Technology

The global agricultural sector is currently undergoing a profound transformation, often characterized as the shift from Agriculture 4.0—defined by automation and data exchange—to Agriculture 5.0, which emphasizes robotics, artificial intelligence (AI), and the creation of cyber-physical systems.[1] At the forefront of this transition is the concept of the "Digital Twin," a virtual replica of physical entities that enables real-time monitoring, simulation, and predictive analytics.[2] Team Mithuna's 'Agri Twin' project exemplifies this technological convergence, deploying an autonomous crop scanning unit powered by the NVIDIA Jetson Orin Nano to bridge the gap between physical field conditions and digital management systems.

The 'Agri Twin' initiative is not merely a robotic exercise; it is a systemic approach to revolutionizing farming operations through advanced technology. As articulated by the project's leadership, the mission extends to transforming traditional farms into highly efficient operations by creating digital twins of biological assets.[3] While the broader scope of Team Mithuna includes dairy farming—optimizing assets from feed crops to precision feeding based on genomic testing—the autonomous crop scanning unit represents the critical data ingestion node for the arable component of this ecosystem.[3] By deploying an edge-AI rover, the system captures granular, plant-level data that informs the digital twin, allowing for the precise measurement of crop health, growth stages, and disease prevalence.

This report provides an exhaustive technical analysis of the 'Agri Twin' crop scanning unit. It dissects the computational architecture centered on the Jetson Orin Nano, the electromechanical challenges of interfacing 1.8V logic with 5V actuation systems, the software pipeline leveraging YOLOv8 and TensorRT for real-time inference, and the navigation algorithms required for autonomous operation. Furthermore, it explores the connectivity framework that synchronizes the physical rover with its digital counterpart, ensuring that the promise of "smart farming solutions where technology and agriculture work hand in hand" is realized.[3]

1.1. The Strategic Imperative of Digital Twins in Agriculture

The adoption of digital twin technology in agriculture addresses critical inefficiencies in traditional crop management. Conventional methods often rely on manual scouting, which is labor-intensive, time-consuming, and prone to human error, potentially leading to delayed detection of pathologies.[4] In contrast, a digital twin provides a dynamic, up-to-date representation of the farm, continuously synchronized with data from field sensors and autonomous units.[6]

The 'Agri Twin' system operates within this paradigm by functioning as a mobile sensor platform. Unlike static IoT sensors that measure environmental parameters (soil moisture, temperature) at fixed points, the autonomous unit traverses the field, capturing high-resolution visual data of the crops. This data is processed at the edge to identify specific pathologies—such as tomato blight or corn rust—and transmitted to the central model.[2] The integration of this real-time data allows the digital twin to simulate future scenarios, such as the spread of a fungal infection based on current humidity levels, and prescribe precise interventions, thereby optimizing resource usage and enhancing sustainability.[1]

1.2. Project Leadership and Technical Vision

The development of such a sophisticated system requires a multidisciplinary approach. Team Mithuna's leadership structure reflects this necessity, combining expertise in data engineering, simulation, and business strategy. Lalit Yagnik, serving as Co-Founder and CTO, brings over three decades of experience in digital transformation, AI, and big data analytics, driving the technical roadmap for revolutionizing the industry.[3] His leadership ensures that the 'Agri Twin' is not just a hardware prototype but a scalable solution capable of integration with cloud technologies.

Supporting this vision is Mohan Babu, the Data Lead, who orchestrates the complex data pipelines required to process the vast amounts of farm information generated by the scanning units. His role is pivotal in transforming raw sensor data into actionable insights that power the digital twin technology.[3] On the simulation front, Xin leads the technical design of farm simulation models. His work ensures that the digital representation accurately mirrors real-world operations, a requirement for valid predictive analytics.[3] This collaborative synergy underpins the 'Agri Twin' project's ability to push the boundaries of AgriTech through innovation, sustainability, and integrity.[3]

2. Computational Core: NVIDIA Jetson Orin Nano Architecture

The efficacy of an autonomous crop scanning unit is fundamentally constrained by its onboard computational capabilities. The system must simultaneously handle visual perception (running deep neural networks), localization (processing odometry and LiDAR data), and path planning, all while operating within a limited power envelope suitable for battery-powered deployment. The NVIDIA Jetson Orin Nano has been selected as the computational engine for the 'Agri Twin', representing a significant leap in edge AI performance compared to previous generations.

2.1. Processing Specifications and Architectural Advantages

The Jetson Orin Nano is engineered to bring transformer-class models and generative AI capabilities to the edge. It delivers up to 67 Trillion Operations Per Second (TOPS) of AI performance, a metric that effectively renders it 80 times more powerful than the original Jetson Nano.[7] This performance density is critical for agricultural robotics, where the complexity of identifying plant diseases in unstructured environments demands substantial neural network throughput.

GPU Architecture:
At the heart of the Orin Nano is the NVIDIA Ampere architecture GPU, featuring 1024 CUDA cores and 32 Tensor Cores.7 The inclusion of Tensor Cores is particularly significant for the 'Agri Twin' application. These specialized execution units accelerate matrix multiplication operations—the foundational math of deep learning—allowing for mixed-precision computing. By leveraging FP16 (half-precision) or INT8 (integer) precision, the Orin Nano can execute object detection models like YOLOv8 with dramatically reduced latency compared to standard FP32 execution.8 This acceleration enables the rover to process video frames at high frame rates (30+ FPS), ensuring that no crop anomalies are missed as the robot moves through the field.
CPU and Processing Efficiency:
The module is equipped with a 6-core Arm Cortex-A78AE v8.2 64-bit CPU, featuring 1.5MB of L2 cache and 4MB of L3 cache.7 This CPU architecture is designed for safety-critical and autonomous applications (denoted by the 'AE' suffix for Automotive Enhanced). It handles the orchestration of the Robot Operating System (ROS), managing the communication between the perception stack (running on the GPU) and the actuation stack (motor controllers). The CPU's operating frequency of up to 1.5 GHz ensures that non-AI tasks, such as reading encoders or managing WebSocket connections, do not become bottlenecks.7
Memory Subsystem:
Agricultural environments are visually complex, requiring high-resolution imagery to detect subtle disease symptoms like early-stage leaf spots. The Jetson Orin Nano addresses this with 8GB of 128-bit LPDDR5 memory, providing a bandwidth of 102 GB/s.7 This high bandwidth is crucial for feeding the GPU with large image tensors without stalling the processing pipeline. It allows the system to support multiple camera streams simultaneously, potentially enabling a 360-degree view of the crop canopy.
Power and Efficiency:
The module operates within a configurable power envelope of 7W to 15W (and up to 25W for the developer kit carrier), making it ideal for mobile robots where battery life is a primary constraint.7 This efficiency allows the 'Agri Twin' unit to operate for extended periods in the

field without requiring heavy, expensive battery packs.

2.2. Carrier Board Implementation and Industrial Robustness

While the Jetson Orin Nano module provides the compute power, its integration into a robotic chassis requires a carrier board to break out the I/O interfaces. For the 'Agri Twin' project, the choice of carrier board dictates the connectivity options for sensors and actuators.

Developer Kit vs. Industrial Solutions:
The standard NVIDIA Jetson Orin Nano Developer Kit serves as the reference implementation. It provides a comprehensive array of I/O, including Gigabit Ethernet, USB 3.2 Gen 2, and a 40-pin GPIO header.7 However, for harsh agricultural environments—characterized by dust, vibration, and temperature fluctuations—industrial carrier boards are often preferred. Solutions like the A603 Carrier Board or the reComputer J4012 offer ruggedized features, such as wide-voltage DC input (9-20V or 9-36V) and M.2 Key E/M slots for expanding storage (NVMe SSDs) and wireless connectivity (Wi-Fi/Bluetooth).9 These boards often include additional industrial interfaces like CAN bus, which is standard in agricultural machinery, potentially allowing the 'Agri Twin' to interface with larger tractors or implements.11
Thermal and Mechanical Considerations:
The 'Agri Twin' unit must operate in outdoor conditions where temperatures can be high. The Orin Nano module typically includes a thermal solution (heatsink and fan). Industrial enclosures, such as the fanless designs offered by some third-party vendors, utilize the chassis itself for heat dissipation, preventing dust ingress—a critical feature for long-term reliability in farming contexts.10

2.3. The GPIO Voltage and Drive Strength Challenge

One of the most critical, yet frequently overlooked, aspects of implementing the Jetson Orin Nano in robotics is the electrical characteristic of its General Purpose Input/Output (GPIO) pins. Understanding this is vital for the 'Agri Twin' project to prevent hardware failure.

1.8V vs. 3.3V Logic:
The Orin Nano System-on-Chip (SoC) natively operates at 1.8V logic levels. To make the 40-pin expansion header compatible with the vast ecosystem of Raspberry Pi accessories (which use 3.3V), the carrier board employs bidirectional voltage-level translators, specifically the TI TXB0108RGYR.12
The "Weak Buffer" Limitation:
The TXB0108 is a "weak buffered" auto-direction sensing translator. It uses an output buffer with an internal series resistor of approximately 4kΩ.12

- **Implication:** This design is intended to drive high-impedance loads (like logic inputs of other chips) but has extremely low drive strength. It is designed to drive capacitive loads of only up to 70pF.[13]
- **The Problem:** Connecting standard motor drivers, relays, or long cables directly to these pins can present a load that exceeds the drive capability. If a connected device has a pull-up or pull-down resistor lower than ~50kΩ, or significant capacitance, the TXB0108 may fail to drive the signal to the required high (3.3V) or low (0V) state. Users frequently

report measuring intermediate voltages (e.g., 1.5V) or seeing oscillation, leading to erratic robot behavior.[13]

Architectural Consequence for 'Agri Twin':
The 'Agri Twin' team cannot reliably drive the actuation systems (motors, servos) directly from the Jetson's GPIO header. Doing so risks signal corruption and potential damage to the carrier board. Instead, the architecture must incorporate intermediate buffer stages or utilize bus-based communication protocols (like I2C) to interface with actuators, as detailed in the subsequent sections on electromechanical actuation.

3. Electromechanical Actuation: Bridging the Logic-Power Gap

The locomotion system of the 'Agri Twin' robot is its means of interacting with the physical world. Navigating crop rows requires precise motor control, typically achieved through differential drive configurations. This section details the specific hardware selection and the engineering solutions required to bridge the high-performance, low-voltage compute of the Jetson Orin Nano with the high-power requirements of DC motors.

3.1. Motor Driver Selection: The L298N Dual H-Bridge

For driving the DC motors of the rover, the L298N Dual H-Bridge driver is a ubiquitous choice in prototyping and light-industrial robotics due to its availability and robustness.

Operational Theory of the L298N:
The L298N integrates two H-Bridge circuits. An H-Bridge is an electronic circuit that switches the polarity of a voltage applied to a load. By closing switches diagonally (e.g., top-left and bottom-right), current flows in one direction; reversing the switches reverses the current.15 This allows the robot to drive motors forward and backward.
Key Specifications:

- **Motor Supply Voltage (Vs):** Supports a wide range from 5V up to 46V, accommodating various battery configurations (e.g., 3S LiPo or 12V Lead-Acid).[16]
- **Logic Supply Voltage (Vss):** Requires a separate 5V logic supply to power the internal gates.
- **Current Capacity:** It can handle up to 2A continuous current per channel (with a peak of 3A), which is sufficient for the gear motors typically used in crop scanning rovers.[18]
- **Voltage Drop:** A significant characteristic of the L298N is the voltage drop across its Darlington transistor outputs, which is approximately 2V. If the battery supplies 12V, the motors will receive only ~10V, a factor that must be accounted for in the power budget.[16]

Control Interface:
The module exposes control pins for each channel:
- **ENA/ENB:** Enable pins. When High, the channel is active. Applying a PWM signal here

controls the speed of the motor.[18]

- **IN1/IN2 & IN3/IN4:** Direction control pins. Setting IN1 High and IN2 Low drives the motor forward; reversing states drives it backward.[15]

3.2. Implementation Architecture: The PCA9685 Solution

As identified in the GPIO analysis, connecting the L298N directly to the Jetson Orin Nano is problematic. The L298N's logic inputs are TTL compatible (ideally 5V), and while they can technically register 3.3V as high, the Jetson's weak TXB0108 drivers may struggle to maintain signal integrity against the input impedance and capacitance of the driver and wiring.[13] Furthermore, the Jetson has limited hardware PWM pins available on the header, complicating speed control for multiple motors.[20]

The Solution: PCA9685 I2C PWM Driver
To resolve these electrical and resource constraints, the 'Agri Twin' architecture utilizes the PCA9685, a 16-channel, 12-bit PWM driver that communicates via I2C.21
Advantages of this Architecture:

1. **Logic Level Isolation:** The PCA9685 communicates with the Jetson over the I2C bus (SDA/SCL). The I2C pins on the Jetson are open-drain and pull up to 3.3V, which is compatible. The PCA9685 itself can be powered by 5V (VCC), and its output PWM signals will be 5V logic.[23] This provides a strong, clean 5V signal to the L298N, matching its optimal logic levels and bypassing the "weak buffer" issues of the Jetson's GPIOs.[22]
2. **Resource Offloading:** Generating high-frequency PWM signals in software (bit-banging) on a non-real-time OS like Linux (Ubuntu) consumes CPU cycles and can be jittery, leading to uneven motor speed. The PCA9685 handles PWM generation in hardware, freeing the Jetson's CPU for complex AI tasks.[23]
3. **Scalability:** This setup requires only two wires (I2C) from the Jetson to control up to 16 motors or servos, allowing for easy expansion (e.g., adding a pan-tilt mechanism for the camera or a robotic arm for sample collection).[22]

3.3. Detailed Wiring and Connection Guide

The robust implementation of the 'Agri Twin' requires precise wiring to ensure signal integrity and power isolation.

Table 1: Jetson Orin Nano to PCA9685 Interconnect

Jetson Orin Nano (Header J41)	PCA9685 Pin	Description	Reference

Pin 3 (SDA)	SDA	I2C Data Line (Bus 1)	[24]
Pin 5 (SCL)	SCL	I2C Clock Line (Bus 1)	[24]
Pin 1 (3.3V)	VCC	Logic Power for PCA9685 Chip	[25]
Pin 6 (GND)	GND	Common Ground	[25]

Note: While the PCA9685 VCC can accept 5V, powering it with 3.3V ensures the I2C logic levels match the Jetson without risk. However, the PCA9685 is generally 5V tolerant on inputs. If the L298N requires strict 5V logic, the PCA9685 VCC should be 5V, but this technically violates the Jetson's 3.3V I2C limit without a level shifter. In practice, many users power PCA9685 VCC with 3.3V, which is sufficient to drive L298N inputs.

Table 2: PCA9685 to L298N Actuation Wiring

PCA9685 Channel	L298N Pin	Function	Logic
PWM 0	ENA	Motor A Speed Enable	PWM Duty Cycle (0-100%)
PWM 1	IN1	Motor A Direction 1	High/Low
PWM 2	IN2	Motor A Direction 2	Low/High
PWM 4	ENB	Motor B Speed Enable	PWM Duty Cycle (0-100%)
PWM 5	IN3	Motor B Direction 1	High/Low
PWM 6	IN4	Motor B Direction 2	Low/High

Power Distribution:
Crucially, the motors draw significant current and introduce electrical noise. They must be powered by a separate high-current source (e.g., a LiPo battery connected to L298N's +12V input). The grounds of the Jetson, PCA9685, and the Battery must be tied together to

establish a common reference frame.16 The PCA9685 also has a "V+" terminal for driving servos; if used, this must also be powered externally, not from the Jetson's 5V rail, to prevent brownouts.22

4. Perception System: Deep Learning for Disease Detection

The core value proposition of the 'Agri Twin' project is its ability to perform autonomous crop scouting. This is achieved through a sophisticated perception stack that utilizes deep computer vision to identify, classify, and localize plant diseases in real-time.

4.1. Neural Architecture: The Evolution of YOLO

The 'Agri Twin' employs the YOLO (You Only Look Once) architecture, specifically the **YOLOv8** iteration, for object detection. This choice represents a strategic balance between accuracy and inference speed, which is paramount for a moving robot.

From YOLOv5 to YOLOv8:
While YOLOv5 established a modular PyTorch foundation, YOLOv8 introduced significant architectural refinements. It utilizes a decoupled head (separating classification and regression tasks) and an anchor-free prediction mechanism.26 This shift reduces the number of box predictions and speeds up Non-Maximum Suppression (NMS), a critical post-processing step.

- **Backbone:** YOLOv8 employs a C2f module (Cross-Stage Partial bottleneck with two convolutions), which improves gradient flow and feature extraction compared to the C3 modules in previous versions.[26]
- **Performance:** In benchmarks, YOLOv8n (Nano) achieves approximately 37.3% mAP (mean Average Precision) on the COCO dataset while maintaining extremely low latency.[26] For agricultural tasks, this translates to the ability to detect small lesions on leaves even when the robot is in motion.

Future-Proofing (YOLOv11 & YOLO26):
The field moves rapidly. Recent developments like YOLOv11 and the proposed YOLO26 offer further efficiency gains. YOLOv11 introduces C3k2 bottlenecks for better feature aggregation, while YOLO26 proposes removing NMS entirely for end-to-end inference.26 Team Mithuna's architecture is likely designed to be model-agnostic, allowing for upgrades to these newer architectures as they mature.

4.2. TensorRT Optimization and Edge Deployment

Running complex PyTorch models directly on an embedded device can be inefficient. To unlock the full potential of the Jetson Orin Nano, the 'Agri Twin' software stack utilizes **NVIDIA TensorRT**, a high-performance deep learning inference optimizer.

The Optimization Pipeline:

1. **Export:** The trained YOLOv8 model (.pt) is exported to an ONNX format and then compiled into a TensorRT engine (.engine).
   - Command: yolo export model=yolov8n.pt format=engine device=0.[27]
2. **Precision Calibration:** TensorRT allows for reduced precision inference. While training occurs in FP32 (32-bit floating point), inference on the Orin Nano is optimized for FP16 or INT8. The Orin Nano's Ampere GPU supports sparse INT8 tensor cores, which can effectively double the throughput compared to FP16 without significant loss in detection accuracy.[7]
3. **Kernel Auto-Tuning:** During the build phase, TensorRT selects the optimal kernel implementations for the specific GPU architecture (Ampere), ensuring maximum utilization of the CUDA cores.[8]

This optimization pipeline enables the 'Agri Twin' to process high-resolution input streams at real-time frame rates (30+ FPS), a necessity for scanning large fields efficiently.

4.3. Agricultural Datasets and Pathology

The intelligence of the system is defined by the data it is trained on. The 'Agri Twin' leverages a combination of public academic datasets and custom field data to cover a wide range of crop pathologies.

Target Crops and Diseases:
The system is trained to recognize specific diseases that cause significant economic loss:

- **Corn (Maize):**
  - *Pathologies:* **Common Rust** (pustules on leaves), **Gray Leaf Spot** (rectangular lesions), and **Northern Leaf Blight** (cigar-shaped lesions).[28]
  - *Dataset:* Derived from PlantVillage and PlantDoc, containing ~4,188 images balanced across these classes.[29]
- **Tomato:**
  - *Pathologies:* **Late Blight** (a devastating water mold), **Early Blight**, **Bacterial Spot**, **Septoria Leaf Spot**, and **Yellow Leaf Curl Virus**.[31]
  - *Dataset:* The PlantVillage dataset provides over 20,000 images of tomato leaves across 10 disease classes and one healthy class.[31]
- **Chili (Pepper):**
  - *Pathologies:* **Leaf Curl**, **Leaf Spot**, **Anthracnose** (fruit rot), and **Whitefly** infestation.[33]
  - *Recent Data:* New datasets (2025) from Bangladesh categorize chili plants into "HUSD" (Healthy, Unhealthy, Seed, Dry) and specific growth stages, enhancing the model's utility for yield estimation as well as disease detection.[34]

The "PlantVillage Gap" and Mitigation:
A critical challenge in agri-robotics is the domain gap between training data and real-world conditions. The PlantVillage dataset, while extensive, consists largely of leaves imaged in

controlled lab settings with uniform backgrounds.36 Models trained solely on this data often fail in the field due to complex backgrounds, variable lighting, and overlapping leaves.
To mitigate this, the 'Agri Twin' project likely employs:

1. **Data Augmentation:** Techniques like random rotation, noise injection, gamma correction, and mosaic augmentation (mixing images) to simulate field conditions.[31]
2. **In-the-Wild Datasets:** Incorporating datasets like **PlantDoc** or **FieldPlant**, which feature images taken in actual plantations with realistic noise and clutter.[37]
3. **Custom Data Collection:** Using the rover itself to collect field data, which is then annotated (e.g., using Roboflow) and used to fine-tune the model.[38]

5. Autonomous Navigation: Mapping and Localization

For the 'Agri Twin' to function as a truly autonomous agent, it must be able to navigate the crop rows without human intervention. This requires a robust navigation stack capable of estimating the robot's position (localization) and building a representation of the environment (mapping).

5.1. Dead Reckoning and Kinematics

The foundational layer of localization is Dead Reckoning (Odometry). This involves estimating the robot's current pose (x, y, θ) relative to a starting point by integrating measurements of its movement over time.[39]

Differential Drive Kinematics:
For a robot with two active wheels and a caster (differential drive), the position update logic is governed by the wheel velocities. If the robot moves with linear velocity v and angular velocity ω over a time step dt:

- $x_{t} = x_{t-1} + v \cos(\theta_{t-1}) dt$
- $y_{t} = y_{t-1} + v \sin(\theta_{t-1}) dt$
- $\theta_{t} = \theta_{t-1} + \omega dt$

Limitations: Dead reckoning is prone to unbounded drift. Small errors in the estimation of θ (heading) accumulate rapidly, leading to large position errors over time.[39] In a muddy field, wheel slippage exacerbates this issue. Therefore, dead reckoning is rarely used in isolation; it is fused with other sensor data.

5.2. Probabilistic Grid Mapping

To correct for drift and enable obstacle avoidance, the 'Agri Twin' constructs a map of its environment. Given the structured but variable nature of crop rows, a **2D Grid Map** (Occupancy Grid) is an effective representation.

Algorithm Mechanics:
The environment is discretized into a matrix of cells. Each cell m_{i} holds a probability

value $p(m_{i})$ representing the likelihood that the cell is occupied by an obstacle (e.g., a crop stalk).40

- **Ray Casting:** The system uses data from a LiDAR or depth camera. For each measurement, a "ray" is traced from the robot's position to the detected obstacle.
- **Update Rule:** Cells along the ray path are updated as "Free" (probability decreases), while the cell at the endpoint is updated as "Occupied" (probability increases).[40] This approach, often implemented using Bresenham's line algorithm, allows the robot to "carve out" free space in the map as it moves.

Gaussian Grid Maps:
More advanced implementations may use Gaussian Grid Maps, where obstacles are represented not just as binary occupied cells but as Gaussian distributions. This accounts for the uncertainty in the sensor measurements, providing a smoother and more robust map for path planning.40
Path Planning:
Once the map is generated, algorithms like A (A-Star)* or Dijkstra are employed to calculate the optimal path from the robot's current position to the next scan target, avoiding cells marked as occupied (crops).41

6. Connectivity and Digital Twin Integration

The 'Agri Twin' is defined by its connectivity. It is not an island; it is a physical tendril of a digital system. The integration of the physical rover with the virtual digital twin relies on robust, real-time telemetry.

6.1. Real-Time Telemetry via WebSockets

To facilitate low-latency monitoring and control, the system architecture eschews standard HTTP polling in favor of **WebSockets**, which provide a persistent, full-duplex communication channel.[42]

Server-Side Implementation (Python on Jetson):
The Jetson runs a WebSocket server (typically using the websockets or asyncio Python libraries). This server captures video frames from the camera using OpenCV, encodes them, and broadcasts them to connected clients.

- **Encoding Efficiency:** Raw video frames are too large for efficient transmission. The system encodes frames into JPEG format to compress them, and then converts the binary data into a Base64 string for transmission over the text-based WebSocket protocol.[44]

Client-Side Implementation (Flutter):
The user interface is a cross-platform mobile application built with Flutter. This allows farmers to monitor the robot from iOS or Android devices.

- **Stream Handling:** The app uses the web_socket_channel package to connect to the

Jetson. A StreamBuilder widget listens for incoming messages.

- **Decoding:** When a Base64 video frame is received, the app decodes it on the fly using base64Decode and renders it using Image.memory, creating a live video feed.[42]

Table 3: Telemetry Protocol Stack

Layer	Technology	Function
Transport	TCP/IP	Reliable packet delivery.
Session	WebSocket	Persistent, bi-directional connection.
Serialization	JSON / Base64	Data formatting for image and sensor strings.
Application	Python (Server) / Flutter (Client)	Logic for capture and display.

6.2. The Digital Twin Feedback Loop

The ultimate goal of this connectivity is the synchronization of the Digital Twin.

1. **Data Ingestion:** The rover detects a disease hotspot (e.g., identifying "Late Blight" in Sector 4).
2. **Model Update:** This data is transmitted to the cloud-based Digital Twin. The virtual model of Sector 4 is updated to reflect the infection status.[6]
3. **Simulation & Prediction:** The Digital Twin simulates the progression of the disease based on current environmental data (humidity, wind) also collected by the rover. It predicts that Sector 5 is at high risk within 24 hours.[1]
4. **Actionable Insight:** The system alerts the farmer or automatically dispatches the 'Agri Twin' (or a spraying drone) to Sector 5 for preventative treatment, closing the loop between physical sensing and digital intelligence.[2]

7. Conclusion

Team Mithuna's 'Agri Twin' project stands as a sophisticated exemplar of Agriculture 5.0, successfully integrating autonomous robotics, edge artificial intelligence, and digital twin technology. The technical architecture is defined by the selection of the **NVIDIA Jetson Orin Nano**, which provides the requisite 67 TOPS of compute power to run advanced **YOLOv8** models for real-time disease detection.

The implementation details reveal a robust engineering approach to common edge-robotics challenges. The utilization of the **PCA9685 I2C driver** effectively bridges the voltage gap between the Orin Nano's 1.8V GPIOs and the 5V logic required by the **L298N** motor drivers, ensuring reliable actuation. The software stack, optimized with **TensorRT**, enables high-speed inference on field-proven datasets like PlantVillage and PlantDoc, while **probabilistic grid mapping** allows for autonomy in unstructured environments.

Ultimately, the 'Agri Twin' transcends the role of a mere data collector. By establishing a real-time, bi-directional link via **WebSockets** to a **Digital Twin**, it empowers farmers with predictive capabilities—transforming reactive crop management into a proactive, data-driven science. As the project evolves, the integration of more advanced models like YOLOv11 and the expansion into multi-robot coordination will likely further cement its position at the cutting edge of precision agriculture.

Works cited

1. Digital Twin Technology in Agriculture - Agritech Future, accessed on December 10, 2025, https://www.agritechfuture.com/wp-content/uploads/2024/01/Digital-Twin-Technology-in-Agriculture.pdf
2. Digital twin-based applications in crop monitoring - PMC - NIH, accessed on December 10, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC11795032/
3. Our Mission - Digital Twin for Dairy Farming, accessed on December 10, 2025, https://agritwin.ai/about
4. Image-based crop disease detection using machine learning - ResearchGate, accessed on December 10, 2025, https://www.researchgate.net/publication/384404905_Image-based_crop_disease_detection_using_machine_learning
5. Smart Ground Robot for Real-Time Detection of Tomato Diseases Using Deep Learning and IoT Technologies - ICCK, accessed on December 10, 2025, https://www.icck.org/article/abs/tscc.2024.593301
6. Application of Digital Twin Technology in Smart Agriculture: A Bibliometric Review, accessed on December 10, 2025, https://www.preprints.org/manuscript/202507.1502/v1
7. Jetson Orin Nano Super Developer Kit - NVIDIA, accessed on December 10, 2025, https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/nano-super-developer-kit/
8. Tutorial - Ultralytics YOLOv8 - NVIDIA Jetson AI Lab, accessed on December 10, 2025, https://www.jetson-ai-lab.com/tutorial_ultralytics.html
9. A603 Carrier Board for Jetson Orin NX/Nano Datasheet V1.0 - Seeed Studio, accessed on December 10, 2025, https://files.seeedstudio.com/products/NVIDIA/A603-Carrier-Board-for-Jetsson-Orin-NX-Nano-Datasheet.pdf

10. NEXCOM unveils rugged AI computer for agriculture & industry - IT Brief Asia, accessed on December 10, 2025, https://itbrief.asia/story/nexcom-unveils-rugged-ai-computer-for-agriculture-industry
11. Agriculture Technology Solutions - Forecr.io, accessed on December 10, 2025, https://www.forecr.io/pages/agriculture
12. Jetson Nano Developer Kit 40-Pin Expansion Header GPIO Usage Considerations - Seeed Studio, accessed on December 10, 2025, https://files.seeedstudio.com/wiki/reComputer/Jetson_Nano_Developer_Kit_40-Pin_Expansion_Header_GPIO_Usage_Considerations.pdf
13. Jetson Nano GPIO example problem - NVIDIA Developer Forums, accessed on December 10, 2025, https://forums.developer.nvidia.com/t/jetson-nano-gpio-example-problem/75547
14. GPIO Input Signal Help (Jetson Nano) - NVIDIA Developer Forums, accessed on December 10, 2025, https://forums.developer.nvidia.com/t/gpio-input-signal-help-jetson-nano/115108
15. L298N Motor Driver: Pinout, Specifications, and Working Explained - AIChipLink, accessed on December 10, 2025, https://aichiplink.com/blog/L298N-Motor-Driver-Pinout-Specifications-and-Working-Explained_572
16. In-Depth: Interface L298N DC Motor Driver Module with Arduino - Last Minute Engineers, accessed on December 10, 2025, https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/
17. L298N - two-channel motor driver - Botland.store, accessed on December 10, 2025, https://botland.store/motor-drivers-ic/60-l298n-two-channel-motor-driver-5903351248945.html
18. L298N Motor Driver Pinout and Wiring Made Simple for Your Projects | ODG, accessed on December 10, 2025, https://www.origin-ic.com/blog/l298n-motor-driver-pinout-wiring-guide/48220
19. Using GPIO Pins on Jetson Orin Nano: Level Shifter Requirements and Compatibility, accessed on December 10, 2025, https://forums.developer.nvidia.com/t/using-gpio-pins-on-jetson-orin-nano-level-shifter-requirements-and-compatibility/338895
20. NVIDIA Jetson Orin Nano GPIO Header Pinout - JetsonHacks, accessed on December 10, 2025, https://jetsonhacks.com/nvidia-jetson-orin-nano-gpio-header-pinout/
21. Instruction - How a L298N H-Bridge is controlled by a PCA9685 servo controller, accessed on December 10, 2025, https://custom-build-robots.com/raspberry-pi-elektronik/16-kanal-pca9685-servo-kontroller-teil-3-ansteuerung-einer-l298n-h-bruecke/8821
22. Actuators - Donkey Car, accessed on December 10, 2025, https://docs.donkeycar.com/parts/actuators/
23. Mastering Servo Control With PCA9685 and Arduino : 8 Steps - Instructables, accessed on December 10, 2025,

https://www.instructables.com/Mastering-Servo-Control-With-PCA9685-and-Arduino/
24. Jetson Nano - Using I2C - JetsonHacks, accessed on December 10, 2025, https://jetsonhacks.com/2019/07/22/jetson-nano-using-i2c/
25. Install Driver PCF9685 Servo Controller in Jetson Nano | by Muhammad Yunus - Medium, accessed on December 10, 2025, https://yunusmuhammad007.medium.com/install-driver-pcf9685-servo-controller-in-jetson-nano-41b42795ab95
26. Ultralytics YOLO Evolution: An Overview of YOLO26, YOLO11, YOLOv8, and YOLOv5 Object Detectors for Computer Vision and Pattern Recognition - arXiv, accessed on December 10, 2025, https://arxiv.org/html/2510.09653v1
27. Deploy YOLOv8 with TensorRT | Seeed Studio Wiki, accessed on December 10, 2025, https://wiki.seeedstudio.com/YOLOv8-TRT-Jetson/
28. Corn or Maize Leaf Disease Dataset for Image Classification - GTS.AI, accessed on December 10, 2025, https://gts.ai/dataset-download/corn-or-maize-leaf-disease-dataset/
29. eligapris/maize-diseases-detection - Hugging Face, accessed on December 10, 2025, https://huggingface.co/eligapris/maize-diseases-detection
30. Enhanced Maize Leaf Disease Detection and Classification Using an Integrated CNN-ViT Model - NIH, accessed on December 10, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC12208913/
31. Tomato Disease Multiple Sources - Kaggle, accessed on December 10, 2025, https://www.kaggle.com/datasets/cookiefinder/tomato-disease-multiple-sources
32. Tomato Leaf Disease Dataset - Mendeley Data, accessed on December 10, 2025, https://data.mendeley.com/datasets/zfv4jj7855
33. Chili Plant Disease Detection - Kaggle, accessed on December 10, 2025, https://www.kaggle.com/datasets/shuvokumarbasak4004/chili-plant-disease-detection
34. Chili Plant Leaf Disease and Growth Stage Dataset from Bangladesh - Mendeley Data, accessed on December 10, 2025, https://data.mendeley.com/datasets/w9mr3vf56s
35. Chili Plant Dataset HUSD BD - Kaggle, accessed on December 10, 2025, https://www.kaggle.com/datasets/shuvokumarbasak4004/chili-plant-dataset-husd-bd
36. Deep learning networks-based tomato disease and pest detection: a first review of research studies using real field datasets - Frontiers, accessed on December 10, 2025, https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2024.1493322/full
37. FieldPlant: A Dataset of Field Plant Images for Plant Disease Detection and Classification With Deep Learning - IEEE Xplore, accessed on December 10, 2025, https://ieeexplore.ieee.org/iel7/6287639/10005208/10086516.pdf
38. Top Tomato Datasets and Models | Roboflow Universe, accessed on December 10, 2025, https://universe.roboflow.com/search?q=class%3Atomato+plant
39. What is Dead Reckoning – Clear Explanation with Python Simulation – Part I,

accessed on December 10, 2025, https://aleksandarhaber.com/what-is-dead-reckoning-clear-explanation-with-python-simulation-part-i/
40. miyilin/PythonRobotics: https://github.com/AtsushiSakai/PythonRobotics.git - GitHub, accessed on December 10, 2025, https://github.com/miyilin/PythonRobotics
41. AtsushiSakai/PythonRobotics: Python sample codes and textbook for robotics algorithms. - GitHub, accessed on December 10, 2025, https://github.com/AtsushiSakai/PythonRobotics
42. Communicate with WebSockets - Flutter documentation, accessed on December 10, 2025, https://docs.flutter.dev/cookbook/networking/web-sockets
43. Building Real-time Apps With Flutter And WebSockets - Bacancy Technology, accessed on December 10, 2025, https://www.bacancytechnology.com/blog/real-time-apps-with-flutter-and-websockets
44. Display Live Video from bytes coming from websocket in Flutter - Stack Overflow, accessed on December 10, 2025, https://stackoverflow.com/questions/71613821/display-live-video-from-bytes-coming-from-websocket-in-flutter
45. Creating a live video streaming application in Flutter | by Mitrajeet Golsangi - Medium, accessed on December 10, 2025, https://medium.com/dscvitpune/creating-a-live-video-streaming-application-in-flutter-43e261e3a5cc
46. How to convert base64 string into the Image with Flutter? - Stack Overflow, accessed on December 10, 2025, https://stackoverflow.com/questions/54412641/how-to-convert-base64-string-into-the-image-with-flutter
47. 🧙 Flutter Tutorial: How to Show Base64 Images from a REST API Like a Pro | by Navin Prince | Medium, accessed on December 10, 2025, https://medium.com/@navinprince.thegr8gossiper/%EF%B8%8F-flutter-tutorial-how-to-show-base64-images-from-a-rest-api-like-a-pro-6707b60f5203