Multi-Agent Proximal Policy Optimization for Network-Aware Drone-based Crop Image Analysis

Andrew Hellman*, Bishwas Wagle*, Sean Peppers†, Vincent Zheng‡, Alicia Esquivel Morel*, Juan Mogollon*, Jianfeng Zhou*, Arunava Roy§, Kannappan Palaniappan*, Prasad Calyam*
*University of Missouri–Columbia, USA
Email: {andrewhellman, bwbgv, ace6qv, jdmcnw, j.zhou, pal, calyamp}@missouri.edu
†Florida Gulf Coast University, USA
Email: speppers5329@eagle.fgcu.edu
‡Stony Brook University, USA
Email: vincent.zheng.1@stonybrook.edu
§University of Memphis, USA
Email: arunava.roy@memphis.edu

***Abstract*—Drone-based applications are transforming precision agriculture by enabling advanced sensing and analysis of crops by collecting high-resolution visual data. However, these applications need to handle dynamic edge network conditions, limited onboard resources, and fixed flight paths, making it challenging to decide when computation tasks should be performed locally on the drone, offloaded to nearby edge servers, or sent to the cloud. In multi-drone deployments, these decisions are inherently coupled: multiple drones make concurrent offloading choices while sharing wireless links and compute resources, where the action of one drone directly impacts the performance of others. In this paper, we propose a network-aware multi-drone-based computation task offloading framework viz., "FieldVision" that can continuously learn and adapt based on real-time system (e.g., battery level and compute) and network (e.g., bandwidth and latency) conditions. Our FieldVision features a Multi-Agent Reinforcement Learning (MARL) approach based on the Centralized Training with Decentralized Execution (CTDE) paradigm to address the coupled and partially observable nature of multi-drone offloading, aiming to learn a decentralized policy that minimizes latency, reduces deadline violations, and conserves energy across the drone swarm. We evaluate FieldVision using representative precision agriculture workloads under a time-varying wireless network model with continuous bandwidth and latency fluctuations. Our experimental results show that FieldVision improves average episode reward by up to 28.6% over single-agent PPO and significantly outperforms heuristic and rule-based baselines in terms of reward stability and deadline reliability, while preserving fully decentralized execution without inter-drone communication.**

***Index Terms*—Multi-agent reinforcement learning, task offloading, drone systems, edge computing, precision agriculture**

I. Introduction

Drone-based applications are transforming precision agriculture by enabling advanced sensing and analysis through high-resolution visual data. Equipped with cameras and onboard processors, drones are increasingly used for compute-intensive vision tasks such as crop health monitoring, plant counting, and anomaly detection, providing farmers with timely and fine-grained insights that are difficult to obtain using ground-based methods [1], [2]. Many of these applications rely on structured imaging workflows, such as field mosaicking [3], where drones follow predefined flight paths (e.g., zig-zag or S-shaped patterns) to systematically capture imagery across large fields. However, these vision-driven applications are computationally intensive and must operate under strict constraints imposed by lightweight onboard hardware and limited energy budgets. As a result, drones must carefully decide when computation should occur locally versus offloaded to edge infrastructure to meet real-time mission requirements.

Agricultural drone missions are further constrained by domain-specific operational requirements, such as altitude selection. Researchers commonly perform higher-altitude flights (e.g., $\sim$50 m) for rapid, field-wide coverage, and lower-altitude flights (e.g., $\sim$15 m) for high-resolution inspection of specific crop regions at the cost of longer flight times and higher energy consumption. As a result, practitioners typically commit to a single altitude and imaging plan before takeoff and perform all analysis offline after flight completion. Such strategies limit their ability to adapt data collection and analysis based on intermediate results. Adaptive imaging—such as an initial high-altitude survey followed by targeted low-altitude inspection—require compute-intensive analytics to run during or immediately after flight under strict timing constraints, where delayed results may miss narrow operational windows for re-flying or targeted inspection. These challenges are further compounded by the large fields and uneven communication coverage, leading to highly dynamic wireless conditions with fluctuating bandwidth, latency, and congestion.

Furthermore, returning to a ground station or laboratory to process data offline is often infeasible due to narrow temporal windows dictated by lighting conditions, varying weather, and agricultural schedules. At the same time, onboard computation is limited by processing capability and energy availability, while shared edge and cloud resources may become congested as multiple drones offload tasks concurrently [4], [5]. These factors render static or threshold-based offloading strategies ineffective, as decisions that are optimal may instantly become suboptimal with network and system dynamics [6], [7].

To address these challenges, prior work has explored rule-

based and multi-metric offloading strategies that evaluate task urgency, link quality, and available bandwidth to guide offloading decisions [8]. While these approaches improve upon fixed heuristics, they rely on handcrafted logic and struggle to generalize under rapidly changing or unseen conditions [6], [7]. More recent learning-based methods leverage reinforcement learning to improve adaptability [9]–[11], but many adopt single-agent formulations or centralized execution models that fail to scale to multi-drone deployments. In multi-drone agricultural missions, task offloading decisions are inherently coupled. Each drone operates with only local observations, yet its offloading choices directly affect shared wireless links and compute resources used by other drones. This decentralized, partially observable, and dynamically coupled setting cannot be effectively addressed using existing approaches.

Thus, there is a need for a scalable, learning-based offloading framework that supports decentralized decision-making and implicit coordination among drones under dynamic network conditions, without inter-drone communication.

A. Our Contributions and Novelty

In this paper, we propose "FieldVision", a learning-based task offloading framework for real-time precision agriculture missions built upon Multi-Agent Reinforcement Learning (MARL). FieldVision enables distributed drones to cooperatively learn offloading policies that minimize latency, reduce deadline violations, and improve mission efficiency under dynamic network conditions. We formulate the offloading problem as a multi-agent Markov Decision Process (MAMDP), where the state captures real-time network metrics, compute usage, and task characteristics, and action space selects the appropriate compute node (onboard, edge, or cloud) for each task. To learn decentralized policies, we adopt Multi-Agent Proximal Policy Optimization (MAPPO) under the centralized training with decentralized execution (CTDE) paradigm. *Although we do not modify the MAPPO algorithm itself, our contribution lies in the system formulation, state/action design, and integration of network-aware dynamics into a cooperative MARL framework tailored for multi-drone edge computing.* A centralized critic captures inter-drone interactions and shared resource contention during training, while each drone executes independently using only local observations during execution. This design supports scalable coordination without requiring inter-drone communication during deployment.

We evaluate the proposed FieldVision framework using representative precision agriculture workloads (e.g., mosaicing, crop counting) under dynamic network conditions with continuous fluctuations in bandwidth and latency. We compare FieldVision against random and single-tier baselines, single-agent Proximal Policy Optimization (PPO) [12], and a multi-metric rule-based decision engine (MMDE), using latency, energy efficiency, and task completion reliability metrics.

The remainder of the paper is organized as follows. Section II reviews related work, and Section III describes the system model and problem formulation. Section IV introduces the proposed offloading framework, while Section V outlines the experimental setup. Section VI reports and analyzes the evaluation results. Finally, Section VII summarizes the key findings, discusses limitations, directions for future work.

II. Related Work

A. Drone-Based Agricultural Imaging Workflows

Drone swarms are increasingly used in mission-critical applications such as precision agriculture, disaster response, and environmental monitoring [13]. In agricultural settings, multiple drones collaboratively perform sensing and analysis tasks such as crop health monitoring, pest detection, yield estimation, and stand counting. These missions generate large volumes of high-resolution visual data that often exceed the onboard compute and energy capacity of individual drones, motivating the use of edge and cloud resources [14], [15].

One representative agricultural imaging workload is aerial image mosaicking, where frames captured during flight are stitched into a global field-level map for downstream analytics. End-to-end pipelines such as DroneZaic [16] support robust mosaicking of aerial video and enable subsequent analysis stages, including vegetation index computation (e.g., NDVI [17], GNDVI, and SAVI [18]) for assessing plant vigor, stress, and growth stages. Other commonly studied workloads include structure-from-motion for canopy reconstruction [19] and vision-based crop counting. Tasks such as crop counting and vegetation index estimation benefit from near-real-time execution to support adaptive decision-making, whereas mosaicking and SfM often require batch-style processing with high compute demands. These workloads exhibit varying computational intensity, data volume, and latency sensitivity, making agricultural drone missions well suited for adaptive computation offloading strategies that balance local processing and remote execution across available compute tiers.

B. Task Offloading and Learning-Based Coordination

Task offloading and resource management have been widely extended on drone systems to reduce latency, conserve energy, and improve mission throughput under resource constraints [20]. Early approaches relied on static or rule-based policies, where offloading decisions used predefined thresholds on metrics such as bandwidth, signal strength, or task size [6]. While simple to deploy, these heuristic methods lack robustness in environments with continuously varying network conditions and workloads.

To improve adaptability, subsequent work proposed multi-metric decision engines that combine factors such as task urgency, link quality, and available bandwidth into weighted scoring models [8]. Although these approaches offer finer controls, they remain deterministic, generalize poorly to stochastic operating conditions, and require careful manual tuning. In multi-drone deployments, independently applying such heuristics can lead to contention at shared communication and compute resources, resulting in unstable offloading behavior.

More recent studies have explored learning-based techniques, particularly deep reinforcement learning (DRL), to enable adaptive offloading decisions based on observed system

dynamics [9]–[11]. Single-agent DRL allow individual drones to learn responsive policies without handcrafted rules, but does not account for inter-drone interactions, leading to conflicting decisions that exacerbate congestion under shared-resource contention. Centralized learning addresses coordination by using global system state into decision-making. While effective, these approaches typically assume centralized execution or continuous access to global state at runtime, but are impractical in large-scale or infrastructure-limited agricultural settings.

MARL has therefore emerged as a promising paradigm for scalable coordination in multi-drone systems. By modeling the problem as MAMDP, MARL enables each drone to learn an individual policy to optimize a shared global objective. The CTDE paradigm allows agents to share information during training while preserving fully decentralized operation at deployment [7]. Algorithms such as MAPPO have demonstrated strong performance in cooperative control tasks under partial observability and shared resource constraints [11]. However, many existing MARL-based offloading frameworks rely on simplified network models or optimize a narrow set of metrics, limiting their applicability to real-world agricultural scenarios [21], [22]. Our work builds on these foundations by evaluating MARL-based decentralized offloading under diverse workloads and network conditions.

III. System Model and Problem Formulation

We consider a fleet of N drones $\mathcal{D}_i$, $i \in 1, \ldots, N$ executing pre-planned flight paths to collect high-resolution imagery over agricultural fields. Each drone generates computation-intensive tasks that can be executed locally or offloaded to edge or cloud resources under dynamic wireless conditions and heterogeneous computes. The objective is to adaptively select offloading decisions that minimize latency and deadline violations while efficiently utilizing shared resources.

A. Task Model

Each drone $\mathcal{D}_i$ generates a sequence of computation tasks for agricultural sensing workloads. Tasks are decomposed into *chunks* for fine-grained scheduling. In this work, a chunk τ_i^k represents the processing of a full aerial video segment captured during flight and is defined by a tuple $\tau_i^k = \{s_i^k, p_i^k, \Delta_i^k, t_i^k\}$, where s_i^k denotes the payload size, $p_i^k \in \{\text{high}, \text{medium}, \text{low}\}$ is the chunk priority, Δ_i^k is the chunk deadline, and t_i^k identifies the parent task to which the chunk belongs. Chunks of the same task share a common deadline, a task is complete once all its chunks are processed. Deadlines are derived from nominal processing and transmission times, scaled by priority, with higher-priority tasks imposing stricter latency constraints to reflect time-sensitive agricultural operations. Once a chunk is generated, the drone must immediately select an execution location before proceeding with subsequent sensing operations.

B. Network Model

We model wireless communication between drones, edge servers, and the cloud as a continuous time-varying stochastic process. For each communication link ℓ, available bandwidth $B_\ell(t)$ evolves according to a noisy sinusoidal model:

$$B_\ell(t) = \bar{B}_\ell + A_\ell \sin(\omega_\ell t + \phi_\ell) + \epsilon_\ell(t) \tag{1}$$

where $\bar{B}_\ell$ is the baseline bandwidth, A_ℓ the amplitude, ω_ℓ the angular frequency, ϕ_ℓ the phase offset, and $\epsilon_\ell(t)$ represents zero-mean noise. We assume the corresponding latency values vary inversely with bandwidth. This model captures smooth fluctuations caused by mobility, interference, and shared spectrum usage, reflecting real-world wireless behavior in agricultural field deployments. We acknowledge that this sinusoidal model is a simplified abstraction and does not explicitly capture effects such as multipath fading, shadowing, or terrain-induced occlusion. However, it provides a controlled, continuously varying environment to evaluate adaptive decision-making under non-stationary network conditions.

C. Compute Model and Offloading Actions

Each chunk may be executed locally on the drone, edge, or in the cloud. Let $a_i^k \in \{0, 1, 2\}$ denote the offloading action selected by $\mathcal{D}_i$ for chunk τ_i^k, corresponding to local, edge, and cloud execution, respectively. The resulting latency $L_i^k(a_i^k)$ is

$$L_i^k(a_i^k) = L_i^{\text{tx}}(a_i^k) + L_i^{\text{comp}}(a_i^k) \tag{2}$$

where L_i^{tx} represents transmission latency depending on network conditions, and L_i^{comp} is the computation latency modeled as $L_i^{\text{comp}}(a_i^k) = \frac{s_i^k}{\rho_{a_i^k}}$, with $\rho_{a_i^k}$ representing target-specific compute throughput. This abstraction captures heterogeneous drone and infrastructure capabilities while remaining scalable.

D. Multi-Agent Markov Decision Process

We model the task offloading problem as a MAMDP: $\mathcal{M} = \langle \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the global state, $\mathcal{A}_i$ is the action space of agent i, $\mathcal{P}\colon \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_N \longrightarrow \Delta(\mathcal{S})$ defines the environment transition dynamics, $\mathcal{R}\colon \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_N \longrightarrow \mathbb{R}$ is a shared reward function, and $\gamma \in (0, 1)$ is the discount factor. During execution, each drone observes only a local partial observation $o_i \in \mathcal{O}_i$, resulting in decentralized decision-making under partial observability.

E. Global State Space ($\mathcal{S}$)

The global state $s_t \in \mathcal{S}$ at any time t is a composition of the local observations of all N agents, such that $s_t = \{s_{1,t}, s_{2,t}, \ldots, s_{N,t}\}$. The state for a single agent $\mathcal{D}_i$, denoted $s_{i,t}$, is composed of its data queue length ($q_{i,t} \in \mathbb{R}_{\geq 0}$), observed bandwidth to edge and cloud targets $b_{i,t} = \{b_{i,t}^{\text{edge}}, b_{i,t}^{\text{cloud}}\}$, and the shared compute load at edge and cloud resources $c_t = \{c_t^{\text{edge}}, c_t^{\text{cloud}}\}$. A full state observation for drone $\mathcal{D}_i$ at time t is therefore given by the tuple $s_{i,t} = \{q_{i,t}, b_{i,t}, c_t\}$.

F. Reward Function ($\mathcal{R}$)

We employ a shared global reward that promotes timely task completion and reduce end-to-end latency, while penalizing for deadline violations and inefficient resource usage across all compute tiers. The reward also includes an energy-aware penalty to discourage excessive onboard computation on

drones. All agents receive the same reward signal r_t, computed from their joint action $\mathbf{a}_t$ with the objective of minimizing a weighted sum of latency (L) and energy consumption (E). We formulate the reward as the negative of this cost function:

$$r_t = -\left(w_L \cdot L(\mathbf{a}_t, s_t) + w_E \cdot E(\mathbf{a}_t, s_t)\right) \tag{3}$$

Here, w_L and w_E are weighting hyper-parameters such that $w_L + w_E = 1$. By sharing a global reward signal, agents are incentivized to make offloading decisions that optimize system-wide performance rather than individual outcomes.

IV. Methodology: Centralized Training with Decentralized Execution

A. FieldVision Architecture Overview

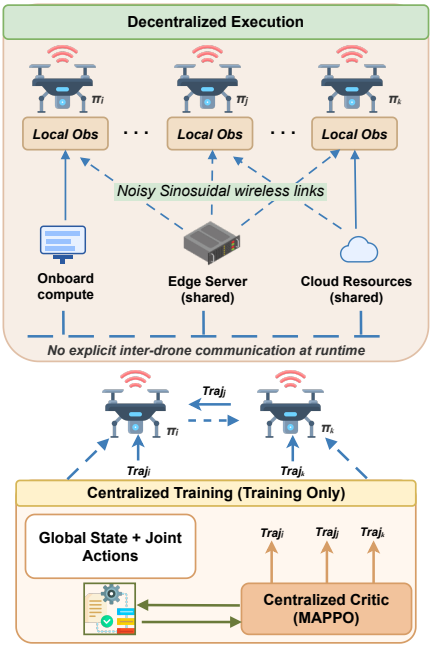

Fig. 1. System architecture of FieldVision under CTDE. Each drone executes a local offloading policy using only local observations, while a centralized critic captures global system state during training to learn coordination-aware policies without runtime communication.

To solve the formulated MAMDP, we employ a MARL approach based on the CTDE paradigm [23]. Under CTDE, agents share global information during training to learn coordinated behaviors, but at execution, each agent operates independently using only locally observables. Fig. 1 shows the FieldVision architecture, decentralized drone execution coordinated through centralized training with a shared critic.

Without centralized coordination, independent learners may compete for offloading resources, causing contention and degraded performance. The CTDE paradigm mitigates this by training a global critic on the joint system state and collective experiences to capture inter-drone interactions. After training, only decentralized policies are deployed, allowing drones to make autonomous real-time decisions while implicitly coordinating through learned behaviors. This eliminates runtime communication overhead and supports coordination across large agricultural drone swarms.

We adopt MAPPO as the core learning algorithm, extending PPO [12] to cooperative multi-agent settings by combining decentralized actor policies with a centralized critic. Each drone $\mathcal{D}_i$ learns an individual policy $\boldsymbol{\pi}_{\theta_i}(a_i \mid o_i)$ that maps local observations o_i to offloading actions, while a shared critic leverages global state during training to capture network dynamics and resource contention. *This* CTDE*-based design enables coordinated behavior while preserving lightweight, communication-free execution at runtime*. Unlike single-agent or independently trained multi-agent, MAPPO explicitly accounts for non-stationarity arising from concurrent policy updates. The centralized critic stabilizes training by providing consistent value estimates as individual agent policies evolve.

B. PPO Surrogate Objective and Training Stability

We optimize each decentralized policy $\boldsymbol{\pi}_{\theta_i}$ using PPO to improve the training stability under multi-agent non-stationarity. In our setting, multiple drones concurrently adapt their offloading strategies while operating over dynamic wireless links and shared compute resources, which can lead to rapid shifts in the underlying reward distribution. PPO addresses this by constraining how much a policy is allowed to change at each update, thereby reducing the risk of destabilizing learning dynamics during centralized training.

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \tag{4}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|o_t)}{\pi_{\theta_{\text{old}}}(a_t|o_t)}$ is the probability ratio between the updated and previous policies, $\hat{A}_t$ is the advantage estimate computed using the centralized critic, and ϵ controls the clipping range that impacts stable learning.

C. Policy and Value Function Parameterization

We parameterize both the decentralized actor policies and the centralized critic using multi-layer perceptrons (MLPs). Each actor takes its local observation as an input, and outputs a probability distribution over offloading actions. The centralized critic receives the aggregated global state and estimates the expected return associated with the joint system configuration. In our implementation, the critic uses a 71-dimensional global state and consists of two hidden layers of size 256. Each actor observes a 12-dimensional local state and is implemented with two hidden layers of size 128. This design balances representational capacity with computational efficiency and is suitable for resource-constrained drone platforms.

V. Experimental Setup

To validate our proposed FieldVision strategy, we designed a custom simulation environment around a representative precision agriculture workload of *crop counting*, chosen for its real-time constraints and moderate computational complexity. The task involves processing high-resolution aerial videos of captured by drones using object detection models to estimate crop density. This task reflects realistic sensing-to-analysis

workflows in crop counting, which requires real-time responsiveness, balanced energy usage, and adaptive offloading under dynamic network conditions. Figure 2 presents the end-to-end execution pipeline for the evaluation experiments.

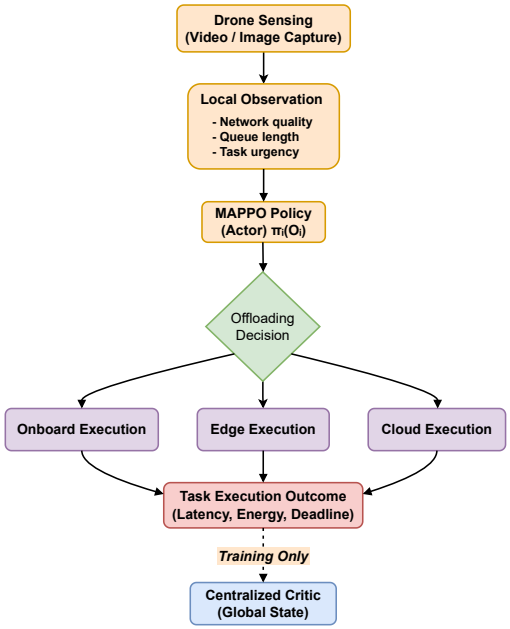

Fig. 2. Experiment Pipeline Overview

A. Simulation Environment

We implement a Gymnasium-based [24] environment to emulate a swarm of $N = 4$ drones operating over a 200×200 acre area. Due to computational constraints of MARL training, we use N = 4 drones as a representative multi-agent setting. However, the CTDE paradigm is inherently scalable, and extending to larger swarm sizes is a key direction for future work. The environment models task generation from each drone, offloading decisions to onboard/edge/cloud compute tiers, time-varying wireless network conditions, and resource contention at shared infrastructure. To capture heterogeneity, we model two *low-capability* drones with reduced onboard compute throughput and higher per-task energy cost, and two *high-capability* drones with stronger throughput and lower per-task energy cost. The environment includes a shared edge server and a cloud server. The edge server provides GPU-accelerated execution with limited parallelism and stochastic background load, modeled as a fixed probability of temporary unavailability. The cloud backend offers highest processing throughput and parallelism but incurs an explicit monetary cost per task, represented as a penalty in the reward function. Table I summarizes compute characteristics.

We represent each simulation episode as a complete flight mission consisting of repeated sensing–decision–execution cycles. At each decision step, every drone observes its local state i.e., task queue, observed bandwidth, and shared resource load, and selects an offloading action. The simulator then executes the chosen actions, computes task completion latency and energy usage, applies deadline penalties when applicable, and returns reward feedback to the learning algorithm.

TABLE I
COMPUTE AGENT CHARACTERISTICS USED IN SIMULATION.

Compute Tier	Throughput	Parallelism	Cost Model
Drone (Low)	5 MB/s	1	High energy drain
Drone (High)	20 MB/s	1	Moderate energy drain
Edge Server	40 MB/s	4	30% busy probability
Cloud Server	100 MB/s	8	Monetary cost per task

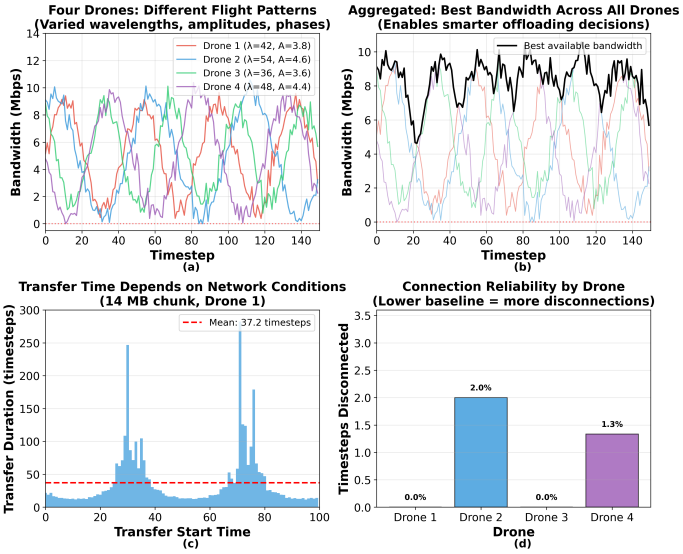

Fig. 3. Time-varying network conditions used in the simulation. (a–b) Per-drone and aggregated wireless bandwidth variations over time. (c–d) Resulting transfer latency variability and connection reliability across drones.

B. Time-Varying Noisy Sinusoidal Wireless Network Model

To reflect continuous network variability in agricultural deployments, we use a noisy sinusoidal model for link bandwidth, making it consistent with the system model in Section III. We model multiple communication links ℓ with distinct characteristics, including Drone–Edge and Drone–Cloud links. For each link ℓ, the available bandwidth:

$$B_\ell(t) = \bar{B}_\ell + A_\ell \sin(\omega_\ell t + \phi_\ell) + \epsilon_\ell(t) \tag{5}$$

where $\bar{B}_\ell$ is the baseline bandwidth, A_ℓ controls fluctuation amplitudes, ω_ℓ defines the fluctuation frequency, ϕ_ℓ the phase offset, and $\epsilon_\ell(t)$ zero-mean noise. Each drone is assigned distinct sinusoidal parameters $(A_\ell, \omega_\ell, \phi_\ell)$ to emulate movement along different flight paths, periodically approaching and receding from cellular infrastructure. Latency is modeled as inversely to bandwidth, and we additionally include a per-link propagation and protocol delay term to capture RTT effects.

Given a task payload size $\text{payload}_{\text{MB}}$, transmission time is computed as -

$$T_{\text{upload}} = \frac{\text{payload}_{\text{MB}}}{B_\ell(t)} + \delta_\ell \tag{6}$$

where δ_ℓ captures fixed RTT/processing overhead for link ℓ. This continuous network model avoids abrupt network

quality switching and enables evaluation under smooth but non-stationary connectivity patterns.

C. Crop Counting Workload Characterization

Each drone continuously generates video-processing tasks that are divided into chunks of size 15–25 MB, with chunk-level deadlines and priorities derived from the parent task, which allows for fine-grained scheduling and partial task progress. The tasks are processed using lightweight object detection pipeline, and counting inference using a lightweight convolutional model. Tasks are generated continuously during each episode at variable rates to make it a suitable test case for evaluating adaptive offloading.

To model compute delay across tiers while keeping training scalable, task computation time is approximated as a linear function of task size: $T_{\text{compute}} = \kappa_{\text{target}} \cdot s$, where s is task size and κ_{target} is a tier-specific coefficient for onboard, edge, or cloud execution. The simulator returns per-task latency, energy consumption, and deadline outcomes after each decision step.

D. Implementation Details and Hyperparameters

We implement MAPPO using Gymnasium with neural network policies parameterized by MLPs. Each centralized critic receives a 71-dimensional global state vector that concatenates observations from all four drones plus 7 global features capturing edge/cloud utilization and in-flight transfer counts, while each actor observes a 12-dimensional local state. Actor and critic networks use two hidden layers with 128 and 256 units, respectively using ReLU activations. We train MAPPO with an actor learning rate of 3×10^{-4}, critic learning rate 5×10^{-4}, discount factor $\gamma = 0.99$, PPO clipping parameter $\epsilon = 0.2$, batch size of 256, and 8 parallel environments with rollouts of 128 steps, totalling 2×10^5 environment steps. All results are averaged over 5 random seeds with 10 evaluation episodes per seed.

E. Reward Instantiation

We instantiate a shared global reward described in Section III that rewards timely completion, energy efficiency, and cooperative behavior across drones. We penalize induced latency, energy drain and excessive cloud usage. Table II summarizes the reward components. The reward coefficients were selected to balance latency, energy consumption, and deadline adherence based on preliminary empirical tuning.

F. Baseline Policies for Comparison

To quantify the benefits of our cooperative MARL approach, we compare FieldVision against the following baselines: **(1) Random:** each drone selects an offloading action uniformly at random. **(2) Onboard-only:** each drone always executes tasks locally, a conservative strategy that avoids network dependence. **(3) Single-Agent PPO:** a non-cooperative reinforcement learning baseline where each drone learns its policy without a centralized critic and without coordination-aware training signals **(4) MMDE:** a rule-based heuristic inspired by prior task offloading work in edge-assisted systems [8]. Collectively, these baselines cover static execution, greedy heuristics, and learning-based approaches with increasing coordination capability. We focus more on **PPO-based** baselines due to their stability and widespread use in MARL settings.

TABLE II
REWARD COMPONENTS AND COEFFICIENTS USED IN SIMULATION.

Reward Component	Value
Base processing reward	+10 per chunk
Offload initiated reward	+8 per transfer
Priority & deadline progress	$2.0 \cdot p + 0.1 \cdot \{d_{\text{rem}}/d_{\text{init}}\}$
Task completion bonus	+40 per task
Cloud execution penalty	-5 per cloud task
Processing delay penalty	$-\min(5, 0.5 \cdot \text{delay})$
Pending chunk penalty	-0.1 per chunk
Battery drain penalty	$-0.5 \cdot (1 - \text{battery\_level})$
Battery empty penalty	-10
Deadline miss penalty	-30
Transfer failure penalty	-15

G. Evaluation Metrics

We evaluate each policy using key performance indicators (KPIs) averaged over multiple simulation episodes and random seeds. Each metric captures a distinct aspect of mission-level performance: **Task Completion Rate (%):** Fraction of generated tasks successfully completed within their deadlines, representing mission reliability. **Average Task Latency (ms):** Mean end-to-end completion time per task, including transmission delay and computation time. **Deadline Violation Rate:** Proportion of tasks that miss deadlines, captureing performance degradation under time-critical workloads. **Offloading Ratio:** Percentage of tasks executed at onboard, edge, and cloud tiers, characterizing policy behavior and resource utilization. **Average Episode Reward:** Mean cumulative reward per episode, which reflects latency, deadline adherence, and energy-aware decision-makingdefined by the reward function.

VI. EVALUATION

In this section, we evaluate FieldVision under the experimental setup described in Section V. Our evaluation addresses the following questions: (i) how cooperative MARL compares against heuristic and learning-based baselines, (ii) whether centralized training improves decentralized execution under shared resource contention, and (iii) how different strategies trade off across completion rate, reward, and offloading behavior under dynamic network conditions. All results are averaged over multiple simulation runs and random seeds. Detailed evaluation figures and analyzes are in Github Appendix.

A. Overall Performance Comparison

We begin by comparing FieldVision against all baseline strategies introduced in Section V-F. While reward captures overall system utility, we also report task completion rate, latency, and deadline violations as primary performance metrics. Table III summarizes performance across these metrics, along with average reward and offloading behavior under dynamic network conditions.

TABLE III
PERFORMANCE COMPARISON ACROSS TASK OFFLOADING STRATEGIES UNDER DYNAMIC NETWORK CONDITIONS.

Method	Completion Rate (%)	Reward (mean±std)	Median Latency	Deadline Violation (%)	Mean Battery Usage (%)	Action Distribution (%) Local	Edge	Cloud
Onboard-Only	83.3	-6693.2 ± 0.0	1.2	16.7	92.0	100.0	0.0	0.0
Edge-Only	21.7	-235.4 ± 71.6	19.9	78.3	0.0	0.0	100.0	0.0
Cloud-Only	21.7	-381.4 ± 60.5	19.9	78.3	0.0	0.0	0.0	100.0
Random	50.3	-184.5 ± 71.2	5.7	49.7	45.3	34.0	33.0	33.0
Round-Robin	55.3	-413.3 ± 53.7	4.1	44.7	51.1	34.0	33.0	33.0
Latency-Greedy	21.8	-381.4 ± 60.5	21.7	78.3	0.0	0.0	0.0	100.0
Multi-Metric	59.3	-5.7 ± 61.4	3.4	40.7	47.5	51.0	49.0	0.0
Multi-Metric (BW)	48.4	-84.7 ± 58.9	4.9	51.6	36.8	37.0	63.0	0.0
Multi-Metric (Urgent)	59.9	1.0 ± 58.0	3.3	40.1	48.1	52.0	46.0	2.0
Single-Agent PPO	66.5	105 ± 45	3.0	33.3	52.6	54.0	46.0	0.0
Centralized PPO	69.7	125 ± 40	2.7	30.3	59.2	59.0	41.0	0.0
MARL (FieldVision)	**68.0**	135 ± 42	**2.9**	**31.8**	**55.7**	**56.0**	**44.0**	**0.0**

Across all evaluated scenarios, FieldVision achieves the highest mean episode reward among decentralized methods while maintaining a high task completion rate (68.0%), a low deadline violation rate (31.8%) and improves mean episode reward by 28.6% over Single-Agent PPO.

Beyond aggregate performance, these results highlight the importance of multi-agent coordination. In non-MARL baselines, offloading decisions are made independently under partial observability, leading to contention for shared wireless and compute resources. This is evident in skewed offloading behavior and increased deadline violations for heuristic and Single-Agent PPO methods, where locally optimal decisions fail to achieve globally efficient outcomes. In contrast, naive strategies such as Random and Onboard-only exhibit clear performance limitations. Random policies suffer from low completion rates and unstable latency due to uncoordinated offloading decisions, while Onboard-only incurs high processing delay and energy cost, resulting in poor reward despite relatively high completion rates.

Rule-based Multi-Metric strategies improve upon naive baselines by incorporating instantaneous system signals. However, these heuristics tend to bias decisions toward specific execution tiers, producing imbalanced offloading behavior and sensitivity to transient network fluctuations. As a result, performance degrades under resource contention, with lower rewards and higher deadline violations compared to learning-based approaches. Learning-based baselines show clear advantages. Single-Agent PPO improves completion rate and reward, but independent learners still exhibit suboptimal offloading patterns under shared edge and cloud contention. This lack of coordination results in increased latency variance and missed deadlines under load. Thus, FieldVision narrows the gap with centralized approaches, demonstrating the effectiveness of cooperative learning under CTDE.

B. Ablation Study: Impact of Cooperative Learning

To isolate the impact of cooperative training, we compare MAPPO against Single-Agent PPO, forming an ablation over coordination awareness rather than network architecture. These results are summarized in Table III and Fig. 4.

Single-Agent PPO learns decentralized policies without coordination-aware training signals. As shown in Fig. 4, this results in lower and more variable episode rewards compared to MAPPO, which benefits from centralized training. MAPPO achieves a higher mean episode reward of approximately 3% than Single-Agent PPO. This gain reflects MAPPO's ability to implicitly learn coordinated offloading behavior that reduces contention at shared resources. These results confirm that centralized training provides measurable coordination benefits even when execution remains fully decentralized and communication-free.

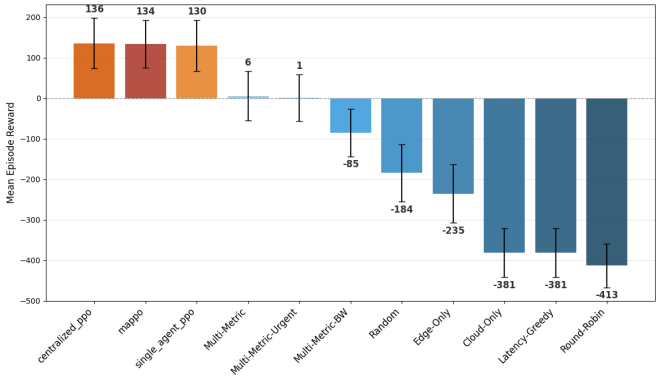

Fig. 4. Ablation study results comparing episode rewards for MAPPO and PPO-based policy variants. Single-Agent PPO lacks coordination-aware training, while Centralized PPO serves as an upper-bound with global execution. MAPPO achieves comparable rewards to centralized execution while preserving decentralized execution.

TABLE IV
QUALITATIVE PERFORMANCE TRADE-OFFS ACROSS TASK OFFLOADING STRATEGIES.

Method	Latency	Deadline Reliability	Reward Stability	Key Trade-off
Onboard-only	Low	Medium	Very Low	Avoids network use at the cost of extreme energy consumption and processing delay
Edge-only / Cloud-only	High	Very Low	Low	Aggressive offloading overloads shared resources, causing severe deadline violations
Random / Round-Robin	Unstable	Low	Low	No adaptation or coordination across network or workload conditions
Multi-Metric	Medium	Medium	Medium	Sensitive to fixed weighting choices and transient network fluctuations
Single-Agent PPO	Medium	Medium	Medium–High	Learns adaptively but lacks coordination awareness under shared resource contention
Centralized PPO	Low	High	High	Requires global state access and centralized execution at runtime
MARL (FieldVision)	**Low**	**High**	**High**	**Balanced decentralized coordination enabled via CTDE**

C. Summary of Performance Trade-offs

Table III presents clear quantitative performance trade-offs across task offloading strategies. Building on these results, Table IV synthesizes the findings by summarizing the dominant qualitative trade-offs across representative strategies.

Heuristic and single-tier baselines exhibit extreme trade-offs as they optimize individual objectives at the expense of overall system performance. Learning-based approaches achieve more balanced behavior, with coordination-aware methods demonstrating improved robustness under shared resource contention. Among decentralized policies, FieldVision consistently maintains favorable trade-offs across all evaluated metrics, without relying on centralized execution or runtime communication.

A detailed qualitative breakdown of metric-level trade-offs, per-variant comparisons and sensitivity analyses, is provided in the Github Appendix

VII. Conclusion

In this paper, we introduced FieldVision, a multi-agent reinforcement learning framework for adaptive computation offloading in drone-based precision agriculture. Using a crop-counting workload as a representative application, we demonstrated that a CTDE strategy based on MAPPO enables drones to make effective offloading decisions under highly dynamic network conditions. Extensive evaluation show that FieldVision consistently outperforms baseline approaches, including Random, Onboard-only, Single-Agent PPO, and multi-metric heuristic offloading strategies. In particular, cooperative MARL policies achieve lower end-to-end latency, reduced energy consumption, and higher system throughput, highlighting the benefits of coordination-aware learning in shared wireless and compute environments. Importantly, the learned policies exhibit stable and scalable behavior, implicitly adapting to network fluctuations and resource contention without requiring explicit inter-drone communication during execution. While evaluated in an agricultural setting, the framework is broadly applicable to other multi-UAV sensing and edge-assisted analytics workloads such as post-disaster damage assessment, traffic and crowd monitoring, wildfire and flood surveillance, and large-scale infrastructure inspection. Looking ahead, we plan to extend FieldVision beyond simulation to real-world experimental platforms such as the AERPAW testbed. This will enable assessment under realistic wireless dynamics, mobility patterns, and hardware constraints. We also aim to incorporate more complex agricultural perception pipelines such as mosaicking and NDVI analysis, to study scalability under multi-modal computation-intensive workloads.

Acknowledgements

The work is supported by National Science Foundation (NSF) CNS core grant No. OAC-2322063. Any opinions, findings and conclusions expressed are those of the author(s) and do not necessarily reflect the views of the U.S. Government or its agencies.

References

[1] A. Zhu, Z. Zeng, S. Guo, H. Lu, M. Ma, and Z. Zhou, "Game-theoretic robotic offloading via multi-agent learning for agricultural applications in heterogeneous networks," *Computers and Electronics in Agriculture*, vol. 211, p. 108017, 2023.

[2] W. Min, A. Khakimov, A. A. Ateya, M. ElAffendi, A. Muthanna, A. A. Abd El-Latif, and M. S. A. Muthanna, "Dynamic offloading in flying fog computing: optimizing iot network performance with mobile drones," *Drones*, vol. 7, no. 10, p. 622, 2023.

[3] D. E. Kharismawati and T. Kazic, "Dronezaic: A robust end-to-end pipeline for mosaicking freely flown aerial video of agricultural fields," *The Plant Phenome Journal*, vol. 8, no. 1, p. e70033, 2025.

[4] K. Salah, P. Calyam, and R. Boutaba, "Analytical model for elastic scaling of cloud-based firewalls," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 136–146, 2016.

[5] M. Sridharan, P. Calyam, A. Venkataraman, and A. Berryman, "Defragmentation of resources in virtual desktop clouds for cost-aware utility-optimal allocation," in *2011 Fourth IEEE International Conference on Utility and Cloud Computing*. IEEE, 2011, pp. 253–260.

[6] A. Sacco, F. Esposito, G. Marchetto, and P. Montuschi, "A self-learning strategy for task offloading in uav networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 4, pp. 4301–4311, 2022.

[7] H. Guo, X. Zhou, Y. Wang, and J. Liu, "Achieve load balancing in multi-uav edge computing iot networks: A dynamic entry and exit mechanism," *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 18 725–18 736, 2022.

[8] A. E. Morel, P. Calyam, and K. Palaniappan, "Task offloading strategies for agricultural uav systems in dynamic network environments," in *2025 21st International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2025, pp. 344–351.

[9] T. Bao, A. Syed, W. S. Kennedy, and M. Erol-Kantarci, "Sustainable task offloading in secure uav-assisted smart farm networks: A multi-agent drl with action mask approach," *IEEE Transactions on Network and Service Management*, 2024.

[10] H. Guo, X. Zhou, J. Wang, J. Liu, and A. Benslimane, "Intelligent task offloading and resource allocation in digital twin based aerial computing networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3095–3110, 2023.

[11] D. Han, Q. Ye, H. Peng, W. Wu, H. Wu, W. Liao, and X. Shen, "Two-timescale learning-based task offloading for remote iot in integrated satellite–terrestrial networks," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10 131–10 145, 2023.

[12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[13] Y. Wang, W. Chen, T. H. Luan, Z. Su, Q. Xu, R. Li, and N. Chen, "Task offloading for post-disaster rescue in unmanned aerial vehicles networks," *IEEE/ACM Transactions On Networking*, vol. 30, no. 4, pp. 1525–1539, 2022.

[14] Y. Zhang, D. J. Love, J. V. Krogmeier, C. R. Anderson, R. W. Heath, and D. R. Buckmaster, "Challenges and opportunities of future rural wireless communications," *IEEE Communications Magazine*, vol. 59, no. 12, pp. 16–22, 2022.

[15] P. Majumdar, S. Mitra, D. Bhattacharya, and B. Bhushan, "Enhancing sustainable 5g powered agriculture 4.0: Summary of low power connectivity, internet of uav things, ai solutions and research trends," *Multimedia Tools and Applications*, pp. 1–45, 2024.

[16] D. E. Kharismawati and T. Kazic, "Maizaic: a robust end-to-end pipeline for mosaicking freely flown aerial video of agricultural fields," *bioRxiv*, pp. 2024–12, 2025.

[17] U. Mahajan and B. R. Bundel, "Drones for normalized difference vegetation index (ndvi), to estimate crop health for precision agriculture: A cheaper alternative for spatial satellite sensors," in *Proceedings of the International Conference on Innovative Research in Agriculture, Food Science, Forestry, Horticulture, Aquaculture, Animal Sciences, Biodiversity, Ecological Sciences and Climate Change (AFHABEC-2016), Delhi, India*, vol. 22, no. 31, 2016, pp. 1–7.

[18] A. Vera-Esmeraldas, S. Pizarro-Oteíza, M. Labbé, F. Rojo, and F. Salazar, "Uav-based spectral and thermal indices in precision viticulture: A review of ndvi, ndre, savi, gndvi, and cwsi," *Agronomy*, vol. 15, no. 11, p. 2569, 2025.

[19] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, "Vggt: Visual geometry grounded transformer," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 5294–5306.

[20] F. Demir, S. Ahmad, P. Calyam, D. Jiang, R. Huang, and I. Jahnke, "A next-generation augmented reality platform for mass casualty incidents (mci)," *Journal of Usability Studies*, vol. 12, no. 4, 2017.

[21] D. Marek, M. Paszkuta, J. Szyguła, P. Biernacki, A. Domański, M. Szczygieł, M. Król, and K. Wojciechowski, "Swarm of drones in a simulation environment—efficiency and adaptation," *Applied Sciences*, vol. 14, no. 9, p. 3703, 2024.

[22] A. E. Morel, M. Powers, K. Keahey, Z. Murry, T. J. Sitzmann, J. Zhou, and P. Calyam, "Floto: Beyond bandwidth-a framework for adaptable, multi-sensor data collection in scientific research," in *International Conference on High Performance Computing*. Springer, 2025, pp. 427–438.

[23] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[24] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. De Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, and A. Kanervisto, "Gymnasium: A standard interface for reinforcement learning environments," *arXiv preprint arXiv:2407.17032*, 2024.