Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Developing a Low Resource Intensive Algorithm for Early Detection of Flying Swarm of Pest Birds

Justice Raphoto Mosoeu

202214553@spu.ac.za

Sol Plaatje University

Lebelo Serutla

Sol Plaatje University

Alfred Mwanza

Sol Plaatje University

Article

Keywords: pest birds, swarm detection, small object detection, motion gating, YOLOv8-nano, CPU-only inference, precision agriculture

Posted Date: July 29th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10360131/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Developing a Low Resource Intensive Algorithm for Early Detection of Flying Swarm of Pest Birds

Justice Raphoto Mosoeu[1*], Mr Lebelo Serutla[1] and Dr Alfred Mwanza[1]

[1*]Department of Computer Science & IT, Sol Plaatje University, South Africa.

*Corresponding author(s). E-mail(s): 202214553@spu.ac.za;
Contributing authors: lebelo.serutla@spu.ac.za; alfred.mwanza@spu.ac.za;

Abstract

Granivorous pest birds, particularly the Red-billed Quelea, cause severe cereal crop losses in sub-Saharan Africa and disproportionately affect small-scale farmers. Traditional deterrence methods are increasingly ineffective, while modern deep-learning detectors often assume costly GPU resources. This paper proposes a low-resource hybrid detection pipeline that uses motion-based preprocessing to gate deep-learning inference. A Gaussian Mixture Model (GMM) and Motion History Images (MHI) continuously monitor motion and trigger YOLOv8-nano inference only when significant motion is detected. Experiments on a custom dataset of 1,380 swarm-annotated images achieved a test mAP@50 of 74.9%, precision of 77.1%, recall of 79.4%, and an average processing rate of 4.59 FPS on CPU-only hardware. CPU usage averaged 79.88% and dropped to 12.5% during static scenes, supporting the efficiency benefits of motion-gated inference for early-warning detection.

Keywords: pest birds, swarm detection, small object detection, motion gating, YOLOv8-nano, CPU-only inference, precision agriculture

1 Introduction

For millions of people across sub-Saharan Africa, food security, income, and livelihoods are closely tied to production of cereal crop such as sorghum, maize, wheat and millet [1]. These essential crops remain under constant threat from granivorous pest birds, especially the Red-billed Quelea, which consumes the same grains that communities depend on for food and income, thus creating a direct conflict between human livelihoods and natural bird feeding patterns [2]. In severely affected areas, pest birds can consume an entire field within minutes and cause crop losses of up to 60% of annual yields [2]. The threat is significant because small-scale farmers contribute a large share of staple crop production in several African countries, making their resilience important to national and regional food security [3].

Historically, farmers relied on labour-based bird scaring methods, typically children or hired workers tasked with shouting, clapping, and throwing objects to chase birds away. This approach was effective largely because it involved continuous human presence, which birds perceive as a threat. However, the socio-economic landscape has changed substantially. Rural-to-urban migration reduces the availability of farm labour; education policies discourage child labour; rising labour costs make dedicated bird scarers financially unsustainable for small-scale farmers; and the physical strain and safety risks associated with long hours under harsh field conditions make continuous guarding impractical [4]. As a result,

farmers remain exposed: they must protect fields but cannot depend on labour-based protection.

At the same time, conventional bird deterrence mechanisms such as scarecrows, reflective tape, and noise cannons have become increasingly ineffective because birds adapt to static stimuli and eventually ignore them [5, 6]. Governments have used control methods such as aerial pesticide spraying and controlled burning of roosting sites, but chemical control is costly, environmentally damaging, and harmful to non-target species [6, 7]. Despite decades of attempts, bird damage remains persistent, demonstrating a practical and technological gap: existing deterrent solutions are often outdated, labour-intensive, environmentally harmful, or technologically inaccessible for small-scale farmers.

1.1 Why Automation Is Necessary: From Detection to Action

A deterrent system is only as effective as its ability to detect bird presence quickly and consistently. Pest birds exploit moments of inattention, and delayed response can lead to catastrophic damage [8]. Therefore, an automated system must (i) detect birds early, (ii) differentiate meaningful threats from noise, and (iii) trigger a deterrent mechanism without human intervention. However, this is non-trivial because pest birds are small, fast-moving, and often distant in imagery; they blend into complex backgrounds such as sky, clouds, and crops; and traditional computer vision approaches struggle to detect small targets reliably [9–11].

1.2 Research Gap

Modern deep-learning object detectors have advanced detection performance in many domains, but they underperform when targets occupy very few pixels and when strong background clutter and environmental variability are present [10, 12, 13]. The literature identifies recurring challenges for small object detection in aerial conditions: birds occupy very few pixels [15, 16]; features are lost through down-sampling [14, 20]; background noise overwhelms small-object features [10, 23]; lighting changes degrade performance [9, 21]; and motion blur affects confidence and consistency [20, 22]. These limitations are especially important for flying swarms captured in real-time footage, where the frame may be mostly empty and targets appear intermittently.

In addition to detection difficulty, many deep-learning systems assume computational resources that are incompatible with low-cost drone platforms and low-power deployment environments. GPU-class hardware increases cost and operational demands, while continuous per-frame inference wastes computation and energy during static scenes [23–25]. This paper identifies three gaps relevant to low-resource agricultural deployment: (1) lack of affordability due to high computational requirements and hardware price [4, 8, 24], (2) computational inefficiency because detectors process every frame even when no birds are present [10, 23], and (3) scarcity of datasets for small bird swarms, which limits training and evaluation consistency.

1.3 Aim and Objectives

The aim of this research is to develop and evaluate a low-resource-intensive early detection algorithm capable of identifying flying swarms of pest birds in static agricultural environments, enabling affordable, autonomous bird-scaring systems suitable for small-scale farmers and deployment on lightweight, CPU-based drone platforms.

The objectives are to: (1) explore algorithms used for detecting swarms of small moving targets; (2) develop a two-phase framework that combines motion-based preprocessing with lightweight deep learning to reduce unnecessary computation in static scenes; (3) evaluate the performance of YOLOv8-nano under constrained computational resources; quantify computational efficiency (CPU usage, memory consumption, inference speed) relative to continuous detection; and identify practical limitations related to dataset size, environmental variability, and deployment constraints.

2 Literature Review

This section reviews technologies and research relevant to detecting swarms of small, fast-moving objects, with a focus on pest birds. It covers the evolution of tracking and detection, challenges of small object detection, motion detection as a computational filter, swarm-level detection principles, and the hybrid rationale used in this work.

2.1 Evolution of Object Tracking and Detection

Object tracking and detection have progressed through classical motion and temporal-template methods, feature-based tracking, deep-learning detectors, and recent advances targeting small object detection and resource efficiency [2, 4, 5, 9, 10, 12, 20, 23]. Early motion-based approaches used background modelling and temporal change to extract moving objects. Gaussian Mixture Models (GMM) are foundational for foreground extraction under illumination variation and scene dynamics [26]. Motion History Images (MHI) provide a compact representation of motion recency and direction, supporting motion pattern recognition without heavy computation [27]. Reviews show these methods remain valuable for low-cost preprocessing but lack semantic understanding and cannot reliably distinguish bird swarms from other motion sources [28].

Before deep learning became dominant, feature-based trackers and descriptors were widely used to follow objects across frames and reconstruct trajectories [23]. Multi-object tracking pipelines combined detector outputs with association to maintain identities through time, but performance deteriorates for small, low-texture objects under occlusion and clutter, which are common in aerial swarm footage [9, 10, 20]. The deep-learning era introduced two-stage detectors that emphasize localisation accuracy and one-stage detectors that emphasize speed [12, 13]. YOLO evolved toward real-time detection efficiency and culminated in YOLOv8 implementations [30]. However, off-the-shelf detectors still struggle with tiny objects because down-sampling reduces discriminative signals needed for small-target localisation [9, 11].

2.2 Small Object Detection Challenges

Detection of small objects remains difficult due to minimal pixel representation, class imbalance, background clutter, and loss of features through pooling/down-sampling [9, 10, 14]. Surveys highlight the difficulty of targets below 32×32 pixels and emphasise that background complexity and environmental variability can dominate weak target cues [9, 11]. While architectural strategies such as multi-scale feature aggregation and specialised losses can improve small-target recall, they often introduce additional computational cost that reduces suitability for edge deployment [10, 16**?**]. For aerial swarm detection, the challenge is amplified by motion blur and changing lighting, which affect confidence and consistency [20, 21].

2.3 Motion Detection as a Computational Filter

Motion-based preprocessing offers an efficiency pathway by filtering frames or regions of interest and reducing unnecessary deep inference [6, 22, 29]. Background subtraction (including GMM), frame differencing, and optical flow are computationally lightweight and effective in static environments [14, 26, 28]. MHIs further compact motion history into a single representation useful for signalling dynamic events [27].Recent bio-inspired work emphasizes motion sensitivity as a key mechanism for detecting small or fast targets [19, 22]. Because deep detectors lack temporal awareness, coupling motion-sensitive components with a deep verifier can improve efficiency and robustness by allocating compute only when motion indicates potential targets [22, 29].

2.4 Swarm Detection in Agriculture and Adjacent Domains

Direct automated detection of pest bird swarms is a niche area, but relevant insights come from UAV deterrence and autonomous detection literature [8, 24]. Swarm detection can be treated as recognising collective dynamic patterns rather than tracking individuals, which is more computationally tractable and aligns with early warning requirements [2, 17, 22, 29]. This collective detection paradigm motivates swarm-level annotation: the operational need is to detect swarm presence early to trigger deterrence, rather than count individuals.

2.5 Synthesis and Rationale for a Hybrid Pipeline

The literature supports a hybrid two-phase approach for balancing efficiency and accuracy in constrained settings [9, 10, 14]. A motion stage

Table 1 Summary of related work

Study	Domain	Approach	Key limitation
Rivadeneira et al. [5]	Agriculture deterrence	Non-vision deterrents	Ineffective long-term; labour-intensive
Bhusal et al. [8]	UAV deterrence	Rule-based UAV control	No vision-based swarm detection
Schiano et al. [24]	Urban deterrence	CNN detection	High computational cost
Chen et al. [21]	Agricultural repellent	DL + laser	Energy-intensive; costly
Tong & Wu [10]	SOD survey	Deep CNN SOD	Poor edge deployability
Wang et al. [20]	UAV aerial detection	Improved YOLOv8	Computationally heavy
Micaelo et al. [6]	Crop protection survey	Review of deterrents	No practical low-resource solution
This research	Small-scale agriculture	GMM+MHI + YOLOv8-nano	Limited dataset; embedded testing pending

using GMM and MHI can run continuously at low cost, while a lightweight YOLOv8-nano verifier is activated only when motion is detected. This reduces redundant inference during static scenes while maintaining semantic discrimination needed to reduce false alarms and support actionable triggering.

2.6 Comparative Summary of Related Work

Table 1 summarises studies in deterrence, small object detection, and resource-aware vision to position this work.

3 Methodology

The proposed methodology is structured to align directly with the objectives of this study and is organised into five main components: (1) dataset curation and annotation, (2) model selection and training, (3) hybrid pipeline integration, (4) preprocessing and optimisation strategies, and (5) system performance evaluation.

Each component plays a specific role in achieving the study's objective of developing a low-resource-intensive early detection framework. Dataset preparation establishes representative swarm data for training; model training enables semantic detection capability; the hybrid pipeline integrates motion gating with deep learning; preprocessing and optimisation strategies reduce computational load; and performance monitoring quantifies both detection accuracy and resource efficiency. Together, these components form the complete implementation framework illustrated in Fig. 3.

3.1 Dataset Curation and Annotation Strategy

A custom dataset of 1,380 images depicting Red-billed Quelea swarms in flight was assembled from publicly available aerial footage on YouTube. The dataset was designed to reflect real environmental variability described in the study: varying lighting as seen in Fig. 1 (Sample swarm scene 2 and 3), background complexity, and swarm density as seen in Fig. 1 (Sample swarm scene 1). This decision was motivated by the need for generalisation to field conditions, where performance is affected by glare, low contrast sky conditions, and other visual variability [10, 18, 21].

Swarm-level annotation justification. The research objective is early swarm detection rather than individual bird counting. This paper defines a swarm as a cohesive group of birds and emphasises that early warning is triggered by swarm presence rather than requiring per-bird localisation. Swarm-level annotation reduces the annotation burden compared to isolating individuals, and it aligns with the operational goal of activating deterrence when a swarm is detected. Annotation was performed using the Roboflow platform, drawing bounding boxes that enclose visible clusters of birds under a single label (`swarm`). The dataset was partitioned into training (80%, 1,104 images), validation (10%, 138 images), and test (10%, 138 images).

3.2 Data Augmentation

To combat overfitting and improve generalisation given the limited dataset size, augmentation was applied through Roboflow as described



Sample swarm scene 1.



Sample swarm scene 2.



Sample swarm scene 3.



Sample swarm scene 4.

Fig. 1 Sample images from the 1,380-image swarm dataset illustrating environmental variation (lighting, background complexity, and swarm density).

in the study. The augmentation pipeline was designed to simulate real-world conditions and included spatial transformations (random rotation ±15°, horizontal flipping, scaling ±20%), photometric distortions (brightness ±25%, contrast ±15%, saturation ±15%), and environmental noise (motion blur up to 5 px) to represent camera movement and fast-moving birds. This strategy teaches invariance to scale, orientation, and lighting changes expected during deployment.

3.3 Model Selection and Training Protocol

YOLOv8-nano was selected because the study identifies it as providing a practical balance between speed and accuracy for low-resource deployment [10, 14, 20]. Transfer learning was used by initialising the model with pre-trained weights (MS COCO) to accelerate convergence and improve feature extraction. Training was performed for 180 epochs with a batch size of 16 and an input size of 640 pixels. Training monitoring followed Ultralytics logging for box loss, classification loss, and distribution focal loss, while evaluation metrics included precision, recall, and mAP@50 [31, 32]. The final model was selected using the best-performing weights on validation metrics and exported in `.pt` format for integration.

3.4 Training Parameter Selection

The study motivates 180 epochs as a convergence choice for this dataset size. Early experimentation suggested that training below 100 epochs led to underfitting and unstable validation performance, particularly reducing recall for small, fast-moving targets. Increasing epoch count allowed the model to adapt more strongly to dataset-specific characteristics, including motion blur, background complexity, and scale variation. Training beyond 180 epochs was avoided due to diminishing gains and increasing risk of overfitting signalled by loss divergence, consistent with the caution emphasised in small object detection literature [9, 10].

3.5 Hybrid Pipeline Integration

The core innovation is the two-stage hybrid detection pipeline described in the study:

Phase 1: Motion detection (GMM + MHI). As illustrated in Fig. 3, a Gaussian Mixture Model (GMM) background subtractor continuously analyses the incoming video feed at a reduced resolution [26, 28]. Consecutive frames are processed to generate Motion History Images (MHI), enabling temporal motion representation [27]. Contour analysis is then performed on the resulting motion mask to estimate the spatial extent of movement within the scene. Only when the detected motion contour area exceeds a tuned threshold does the pipeline activate the subsequent verification phase. This design ensures that deep inference is not executed during static or low-activity scenes, thereby conserving computational resources [25].

Phase 2: Swarm verification (YOLOv8-nano). Once motion is confirmed in Phase 1, the corresponding frame is passed to the YOLOv8-nano model for semantic verification [30]. This conditional activation mechanism ensures that deep-learning inference is performed only when necessary. The model then predicts bounding boxes around detected swarm regions, and upon positive detection, the system generates an audio alert. By coupling motion-based gating with



Fig. 2 Phase 1: Motion detection (GMM + MHI).

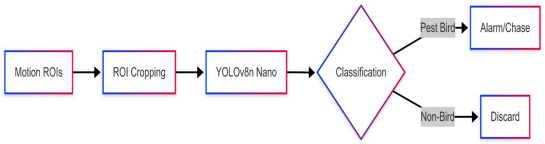


Fig. 3 Phase 2: Swarm verification. Motion gating is used to trigger YOLO inference only during motion events (see Fig. 2).

lightweight object detection, the framework aligns computational effort with scene dynamics while maintaining detection capability [9, 10].

3.6 Preprocessing and Resolution Pipeline

To balance accuracy and computational efficiency, the study defines a resolution-aware pipeline [9, 10]. The input video has a native resolution of 1920×1080. For motion detection (GMM and MHI), each frame is downscaled to 640×480 to reduce cost. When motion triggers inference, the corresponding higher-resolution frame is passed to YOLO and resized to 640×640 for model input [20]. This preserves details for classification only when necessary while keeping continuous monitoring lightweight [25].

3.7 System Performance Monitoring

System performance was evaluated along two axes. Detection accuracy used standard object detection metrics (precision, recall, mAP@50) on the held-out test set [31, 32]. Computational efficiency was measured during video processing using the `psutil` library, which logged CPU utilisation (%), RAM consumption (MB), execution behaviour, and average FPS, which are commonly used indicators of runtime performance in real-time computer vision systems [23, 25]. The system displayed these values (performance metrics) in real time on the user interface while simultaneously logging them for post-experimental analysis.

3.8 Algorithm Optimisation

Following initial testing, the study notes several optimisations that were implemented for CPU performance: input resolution scaling, frame skipping (processing every third frame during high-motion states), and motion threshold tuning to minimise false triggers from minor environmental movement such as cloud motion and camera shake [9, 10]. These optimisations introduce a trade-off where detection continuity may reduce for very fast or dispersed swarms, but they are necessary to achieve viable processing speed and reduce wasted computation [20, 23, 25].

4 Experimental Setup

Experiments evaluated the hybrid system under CPU-only constraints to reflect low-resource deployment feasibility [25, 33]. Testing targeted two main objectives described in the study: (i) evaluating computational efficiency in terms of CPU, memory, and FPS, and (ii) assessing detection accuracy for swarm presence under varying motion and lighting conditions [9, 10, 20].

4.1 Overview of the Experimental Framework

The experimental design was constructed to meet two linked constraints found throughout the problem setting in the study. First, the system must provide reliable early detection for flying swarms in static agricultural environments where targets are small and visually ambiguous [9, 10]. Second, the system must operate under low-resource conditions to remain accessible and deployable [25, 33, 34]. Therefore, the design is not only about achieving accuracy but about managing computation intentionally through architectural decisions rather than relying on expensive hardware acceleration.

4.2 Experimental Setup and Hardware Specifications

All testing and benchmarking were conducted on a Lenovo ThinkPad T490s laptop to simulate low-cost edge constraints and establish a conservative baseline [33, 34]. The device specifications were: Intel Core i5-8365U CPU (1.60 GHz, 4 cores, 8 threads), integrated Intel UHD Graphics 620 (no

dedicated GPU used), 16 GB RAM, and Windows 11 Pro (64-bit). The software stack was implemented in Python 3.8, using OpenCV for video I/O, motion detection, and image processing, and the Ultralytics implementation of YOLOv8 for model training and inference [30]. Training was performed in Google Colab to use its temporary Tesla T4 GPU for training acceleration; however, the deployment evaluation in this paper is explicitly CPU-only to reflect the intended low-resource deployment baseline [25].

4.3 Evaluation Metrics

Detection performance was evaluated using precision, recall, and mAP@50 (IoU threshold 0.5) [31, 32]. Precision reflects the proportion of correct positive predictions and therefore the false alarm behaviour. Recall reflects the proportion of true swarms detected and is important for early warning where missed detections are costly. mAP@50 summarises detection accuracy at the set IoU threshold across recall levels. Computational efficiency was evaluated using FPS (end-to-end processing rate), CPU utilisation, and RAM usage logged through `psutil` during video processing [23, 25].

5 Results

This section reports the results obtained from evaluating the developed algorithm. Results are organised into detection performance, training behaviour, and computational resource utilisation, consistent with the study.

5.1 Detection Performance Under Different Conditions

The study summarises observed system behaviour across four representative scenarios. Table 2 reproduces these qualitative outcomes.

The results suggest that detection performance is strongly influenced by swarm density and environmental visibility conditions [9, 10]. Dense swarm formations produced the most reliable detections because the collective motion of many birds creates stronger motion cues and clearer visual patterns that the detection model can recognise. In contrast, sparse or scattered flocks produced occasional missed detections because the motion signatures were weaker and individual birds occupied fewer pixels within the frame, making them harder to distinguish from background noise [9, 20].

Table 2 Sample detection outcomes

Scenario	Observation	Detection outcome
Dense swarm formation	High contrast, collective motion	Accurate detection
Sparse/scattered flock	Lower motion coherence	Occasional missed detections
Bright sunlight/glare	Background confusion	Increased false positives
Low light/dusk	Reduced visibility	Fewer detections, lower confidence

Environmental lighting conditions also affected system behaviour. Bright sunlight and glare sometimes introduced background artefacts that the model interpreted as potential targets, increasing the likelihood of false positive detections. Conversely, low-light or dusk scenarios reduced the visibility of swarm structures, which lowered detection confidence and reduced the number of successful detections [21].Overall, these observations indicate that the system performs most reliably when swarm motion is cohesive and visual contrast between the swarm and the background is sufficiently strong [10], as illustrated by the swarm development sequence in Fig. 4.



Fig. 4 Temporal sequence of system response during swarm development: (a) Time 1 - Swarm takes flight with negligible motion signature, tracking unactivated. (b) Time 2 - Cohesion remains low as the pattern forms, swarm undetected, tracking unactivated. (c) Time 3 - Swarm partially organized but lacks sufficient visual contrast/coherence for triggering detection. (d) Time 4 - Cohesive swarm motion satisfies gating thresholds, resulting in a successful bounding-box verification and system alert.

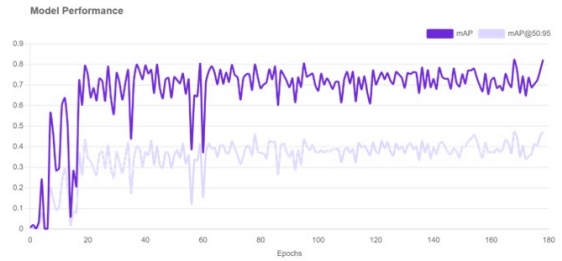


Fig. 5 YOLOv8-nano training and validation loss metrics over 180 epochs.

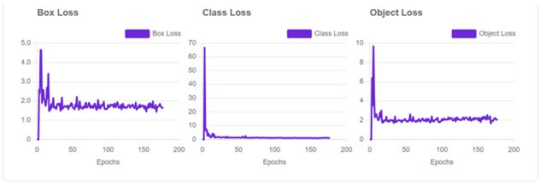


Fig. 6 Model performance metrics through the training process.

Table 3 Validation and test metrics

Metric	Validation set	Test set
mAP@50	82.6%	74.9%
Precision	81.6%	77.1%
Recall	86.4%	79.4%

5.2 Training Performance and Convergence

The study notes that training was conducted over 180 epochs. As illustrated in Fig. 5, the Box Loss decreased rapidly during the early training stages and stabilised around epoch 50, indicating that bounding box localisation was learned early. The Class Loss and Object Loss, shown within Fig. 6, also converged steadily, reflecting stable optimisation behaviour for the single `swarm` class.

The metric trends presented in Fig. 6 further show that mAP@50 fluctuated between 0.65 and 0.80 during training, suggesting sensitivity to dataset variance such as lighting changes and background clutter [31, 32].

5.3 Validation and Test Set Performance

Table 3 reproduces the validation and test metrics reported in the study.

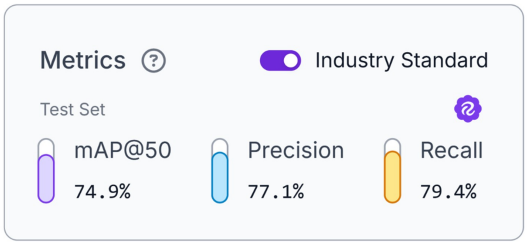


Fig. 7 Ultralytics HUB interface showing final model performance.

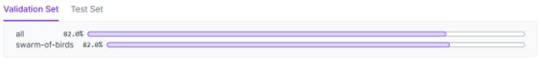


Fig. 8 Confusion matrix for the trained YOLOv8-nano model on the test dataset.

Table 4 System resource summary

Metric	Average	Peak	Minimum
CPU usage (%)	79.88	100	12.50
RAM usage (MB)	12200	12321	12123.93
Frame rate (FPS)	4.59	8.22	1.27

The test mAP@50 of 74.9% indicates good overall accuracy under unseen data, with precision (77.1%) indicating most detections were correct and recall (79.4%) indicating most swarms were detected. The study interprets the drop from validation to test as evidence of generalisation challenges under environmental variability, including swarm density variation, background movement, and resolution differences, consistent with small object detection literature [10, 20, 21].

5.4 System Performance and Resource Utilisation

The study notes that computational performance was measured using `psutil` during runtime, tracking CPU usage, RAM usage, and FPS [25]. Table 4 reproduces the system resource summary.

The study notes that CPU usage peaked at 100% during intense motion and dropped substantially during static scenes, confirming the conditional activation mechanism reduces redundant deep inference [10, 25]. The average FPS (4.59) is interpreted as acceptable for early detection, where timely awareness is prioritised over

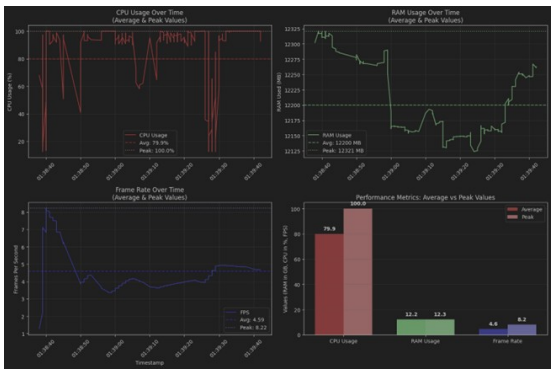


Fig. 9 Detection output of the integrated system.

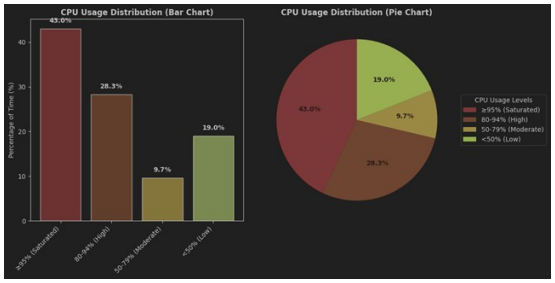


Fig. 10 Performance graphs tracking CPU usage (%) and RAM consumption (MB).

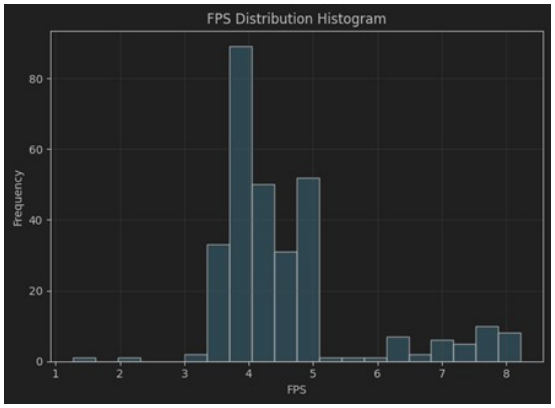


Fig. 11 FPS rate logged over the processing session.

high-frequency tracking [24]. Figures 9–12 illustrate representative outputs and runtime profiling trends reported in the study [20].



Fig. 12 Correlation matrix of system metrics (CPU, RAM, FPS).

5.5 Comparison Context and Dataset Considerations

The study explains that direct quantitative comparison with existing bird detection studies is challenging due to the limited availability of standardised public datasets for agricultural bird swarm detection and because related studies often rely on proprietary datasets or different environmental conditions [21, 24]. In this context, achieving a test mAP@50 of 74.9% with YOLOv8-nano under CPU-only constraints demonstrates that useful accuracy can be achieved under strict computational limits. The study emphasises that the goal is not to outperform GPU-based systems but to demonstrate feasible early detection in constrained settings. The motion-gated design also changes how FPS should be interpreted, because computation is allocated dynamically according to motion events rather than continuously.

6 Discussion

This section interprets and contextualises the results with respect to the study's objectives and literature. Discussion is organised according to detection performance, generalisation, computational efficiency, design trade-offs, deployment feasibility, and objective/hypothesis alignment, following the study.

6.1 Interpretation of Detection Performance

The study interprets the detection performance as reflecting known trade-offs in small object detection under aerial imagery constraints. The achieved test mAP@50 of 74.9% and recall of 79.4% indicate the model generally detected swarms under varied conditions despite minimal pixel footprint and high intra-class similarity [9, 10, 20]. Importantly, the study argues that recall is particularly significant for early warning systems: false negatives are more detrimental than false positives because a missed swarm can translate directly to unmitigated crop damage [8]. From this perspective, the model's sensitivity aligns with the application requirement of early detection.

The study also reports a performance reduction from validation to test metrics, suggesting environmental variability affects generalisation [20, 21]. Challenging conditions such as glare, low contrast skies, and motion blur reduced confidence and consistency [14]. The study attributes this to both dataset size limitations and visual homogeneity between birds and background under certain lighting conditions, aligning with broader literature that emphasises background complexity and illumination sensitivity as persistent issues in small object detection [10].

6.2 Interpretation of Computational Performance

The study interprets computational performance as evidence that the hybrid architecture is effective. Continuous deep inference is computationally expensive, but motion-based pre-filtering reduces unnecessary inference operations [22, 29]. Periods of low or no motion resulted in reduced CPU utilisation, demonstrating the system adapts load to scene dynamics. The study emphasises that the observed average FPS of approximately 4–5 must be interpreted in application context. Early swarm detection prioritises timely awareness and triggering rather than frame-by-frame tracking, and therefore the observed FPS is argued to be sufficient to activate deterrence before substantial crop damage occurs [6, 24].

The study also reports temporary FPS drops during dense motion due to increased YOLO activation and scene complexity [20]. This is interpreted not as a failure but as a direct consequence of allocating computation power when motion indicates potential swarm activity, consistent with the system's design philosophy of spending resources when it matters most.

6.3 Design Trade-offs and System Behaviour

The study stresses that the system does not aim to outperform high-end GPU detectors on raw speed or peak precision [20, 24]. Instead, it aims to balance accuracy and efficiency through architecture rather than brute-force computation [22]. Conditional activation explains robust static-scene behaviour and controlled degradation under complex conditions. This behaviour aligns with the objective of enabling affordable autonomous detection in constrained environments [6, 8].

6.4 Deployment Considerations and Embedded Constraints

The evaluation used CPU-only execution as a baseline to simulate low-resource environments [25]. The study notes that embedded platforms (e.g., Raspberry Pi-class systems or similar) introduce constraints such as limited memory bandwidth, camera I/O bottlenecks, and thermal throttling under sustained workloads [33, 34]. These factors can influence real-time performance, power consumption, and long-term stability [25]. While CPU-based evaluation provides a conservative proof-of-concept, deployment validation requires testing on representative embedded hardware to analyse sensor I/O latency, thermal behaviour, and energy efficiency in real field conditions [35]. Nevertheless, the study argues that conditional activation reduces sustained computational load, which is expected to mitigate thermal and power challenges on embedded platforms and improve deployment feasibility for agricultural monitoring applications [36, 37].

6.5 Alignment with Objectives and Hypothesis

The study concludes that the study objectives were met: the hybrid motion-aware detection algorithm was designed, implemented, and evaluated; and its accuracy and efficiency were quantified.

The quantitative hypothesis target of ≥25 FPS was not achieved on CPU-only hardware. However, the study emphasises that the core intention of the hypothesis—minimising computation while maintaining robust detection—was validated by the conditional activation mechanism. CPU utilisation dropping to 12.5% during static scenes provides evidence that the system reduces active deep inference time substantially, supporting feasibility for constrained deployments even at sub-real-time FPS. The study recommends future testing on edge-AI platforms to quantify hardware-specific behaviour and scalability under real drone integration.

7 Conclusion

This paper developed and evaluated a low resource intensive algorithm for early detection of flying swarms of pest birds, motivated by severe cereal crop losses and the decline of effective labour-based deterrence in sub-Saharan Africa [2, 4]. The study proposed a hybrid two-phase pipeline combining motion-based preprocessing (GMM and MHI) with a lightweight YOLOv8-nano verifier. A custom dataset of 1,380 swarm-annotated images was curated from aerial footage and used to train and evaluate the detector. The final system achieved a test mAP@50 of 74.9%, precision of 77.1%, and recall of 79.4%, demonstrating useful early-warning detection capability under constrained conditions.

From a computational perspective, CPU-only evaluation demonstrated an average processing rate of 4.59 FPS, with CPU utilisation averaging 79.88% and dropping to 12.5% during static scenes. These findings support the central design claim of the study: motion gating reduces redundant deep-learning inference by allocating compute only when motion events occur.

The study also acknowledges limitations. The dataset size and environmental diversity constrain generalisation under extreme lighting and background conditions. The achieved FPS is below the initially stated 25 FPS target for real-time processing; however, early detection does not require high-frequency tracking, and the achieved FPS remains functionally sufficient for triggering deterrence. The current system detects swarm presence but does not yet perform swarm behaviour analysis such as density estimation or trajectory modelling.

Recommendations include field testing with prototype drones in affected agricultural areas, coupling detection with automated deterrent mechanisms, expanding and diversifying the dataset, and exploring further model optimisation (pruning, quantisation, distillation) to improve speed on cheaper embedded hardware. Future work may also investigate multi-modal sensing such as thermal imaging to improve reliability under low light and integrate temporal tracking methods to improve detection continuity.

Acknowledgements. The authors acknowledge the support and guidance of Mr Lebelo Serutla and Dr Alfred Mwanza throughout this research. Appreciation is also extended to the Department of Computer Science & IT at Sol Plaatje University for providing an environment that supported this work. The authors further acknowledge the use of publicly available aerial footage used to curate the dataset and the tools used for annotation and experimentation as described in this paper.

Declarations

- **Funding:** Not applicable.
- **Conflict of interest/Competing interests:** The authors declare no conflict of interest.
- **Ethics approval:** Not applicable.
- **Consent to participate:** Not applicable.
- **Consent for publication:** Not applicable.
- **Availability of data and materials:** The custom swarm dataset and trained model weights are available from the corresponding author on reasonable request.
- **Code availability:** Not applicable.
- **Authors' contributions:** J.R.M. contributed to conceptualisation, methodology, experiments, data analysis, and writing—original draft. L.S. contributed to supervision, methodology guidance, and writing—review and editing. A.M. contributed to conceptualisation, supervision, project administration, and writing—review and editing. All authors reviewed and approved the final manuscript.

References

[1] Sorghum Trust: Sorghum Cluster Initiative Report. Sorghum Trust, South Africa (2023)

[2] Hiron, M., Rubene, D., Mweresa, C.K., Ajamma, Y.U., Owino, E.A., Low, M.: Crop damage by granivorous birds despite protection efforts by human bird scarers in a sorghum field in western Kenya. Ostrich **85**(2), 153–157 (2015)

[3] Elliott, C.C., Mtobesya, B.N., Cheke, R.A.: Alternative approaches to Red-billed Quelea management: mass-capture for food. Ostrich **85**(1), 31–37 (2014)

[4] Touch, V., et al.: Small-scale farmers' challenges and opportunities: implications for agricultural production, environment and food security. J. Environ. Manage. **370**, 122536 (2024)

[5] Rivadeneira, P., Kross, S., Navarro-Gonzalez, N., Jay-Russell, M.: A review of bird deterrents used in agriculture. In: Proceedings of the Vertebrate Pest Conference, vol. 28 (2018)

[6] Micaelo, E.B., Lourenço, L.G.P.S., Gaspar, P.D., Caldeira, J.M.L.P., Soares, V.N.G.J.: Bird deterrent solutions for crop protection: approaches, challenges, and opportunities. Agriculture **13**(4), 774 (2023)

[7] Cheke, R.A.: Forecasting movements and breeding of the Red-billed Quelea bird in southern Africa and improved control strategies. Final Technical Report, Natural Resources Institute, University of Greenwich (2003)

[8] Bhusal, S., Karkee, M., Bhattarai, U., Majeed, Y., Zhang, Q.: Automated execution of a pest bird deterrence system using a programmable unmanned aerial vehicle (UAV). Comput. Electron. Agric. **198**, 106972 (2022)

[9] Liu, Y., Geng, L., Zhang, W., Gong, Y., Xu, Z.: Survey of video based small target detection. J. Image Graph. **9**(4), 122–134 (2021)

[10] Tong, K., Wu, Y.: Deep learning-based detection from the perspective of small or tiny objects: a survey. Image Vis. Comput. **123**, 104471 (2022)

[11] Huang, Z., et al.: A review of small object detection methods in computer vision. Pattern Recognit. **139**, 109392 (2023)

[12] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of CVPR, pp. 779–788 (2016)

[13] Jiao, L., et al.: A survey of deep learning-based object detection. IEEE Access **7**, 128837–128868 (2019)

[14] Zhenghua, Z., Ling, J., Qingqing, H.: A survey of small object detection based on deep learning. In: Proceedings of the 8th IIAE International Conference on Intelligent Systems and Image Processing, pp. 45–52 (2021)

[15] Shao, C., Huang, J., Fan, X., Wang, X., Shi, P.: A bird target detection model designed for substation scenarios. Sci. Rep. **15**(1), 22829 (2025)

[16] Xing, C., Qu, C., Zhang, P., Qin, X.: An efficient method for monitoring small bird targets in wetland environments based on object detection. Sci. Rep. **16**(1), 46593 (2026)

[17] Raza, A., Hanif, F., Mohammed, H.A.: Analyzing the enhancement of CNN-YOLO and transformer based architectures for real-time animal detection in complex ecological environments. Sci. Rep. **15**(1), 26645 (2025)

[18] Mao, X., Chow, J.K., Tan, P.S., Liu, K.F., Wu, J., Su, Z., Ooi, G.L., Pang, C.C., Wang, Y.H.: Domain randomization-enhanced deep learning models for bird detection. Sci. Rep. **11**(1), 639 (2021)

[19] Al-Zadjali, N.S.H., Balasubaramanian, S., Savarimuthu, C., Rances, E.O.: Bio-inspired motion detection models for improved UAV and bird differentiation: a novel deep learning framework. Sci. Rep. **15**(1), 15521 (2025)

[20] Wang, G., et al.: UAV-YOLOv8: a small-object-detection model based on improved YOLOv8 for UAV aerial photography scenarios. Sensors **23**(16), 7190 (2023)

[21] Chen, Y.C., Chu, J.F., Hsieh, K.W., Lin, T.H., Chang, P.Z., Tsai, Y.C.: Automatic wild bird repellent system that is based on deep-learning-based wild bird detection and integrated with a laser rotation mechanism. Sci. Rep. **14**(1), 15924 (2024)

[22] Zheng, J., et al.: Motion feature extraction using magnocellular-inspired spiking neural networks for drone detection. Front. Comput. Neurosci. **19**, 1452203 (2025)

[23] Carranza-García, M., Torres-Mateo, J., Lara-Benítez, P., García-Gutiérrez, J.: On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data. Remote Sens. **13**(1), 89 (2020)

[24] Schiano, F., Natter, D., Zambrano, D., Floreano, D.: Autonomous detection and deterrence of pigeons on buildings by drones. IEEE Access **10**, 1745–1755 (2021)

[25] Teerapittayanon, S., McDanel, B., Kung, H.T.: Distributed deep neural networks over the cloud, the edge and end devices. IEEE Trans. Ind. Inform. **14**(7), 3177–3186 (2017)

[26] Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: Proceedings of CVPR, pp. 246–252 (1999)

[27] Davis, J., Bradski, G.: Motion segmentation and pose recognition with motion history gradients. In: Proceedings of the IEEE Workshop on Human Motion, pp. 246–251 (2000)

[28] Elharrouss, O., Abbad, R., Tairi, H.: A review of background subtraction techniques. J. Vis. Commun. Image Represent. **77**, 103084 (2020)

[29] Xu, M., et al.: STMDNet: a lightweight directional framework for motion pattern recognition of tiny target. arXiv preprint arXiv:2501.13054 (2025)

[30] Jocher, G., Chaurasia, A., Qiu, J.: YOLOv8: Cutting-Edge Real-Time Object Detection. Ultralytics Technical Report (2023)

[31] Rodda, S.R., Raja, K.B.: Performance evaluation metrics for object detection algorithms in computer vision. J. Image Graph. **9**(3), 104–112 (2021)

[32] Padilla, R., Netto, S.L., da Silva, E.A.B.: A survey on performance metrics for object-detection algorithms. In: Proceedings of IWSSIP, pp. 237–242 (2020)

[33] Tsouros, D.C., Bibi, S., Sarigiannidis, P.G.: Review on UAV-based applications for precision agriculture. Information **10**(11), 349 (2019)

[34] Awais, M., Bibi, N., Habib, S., Nazir, M.: Deep learning-based drone surveillance systems: a review. Comput. Electr. Eng. **100**, 107832 (2022)

[35] Tokekar, P., Hook, J., Mulla, D., Isler, V.: Sensor planning for a symbiotic UAV and UGV system for precision agriculture. IEEE Robot. Autom. Lett. **1**(1), 339–346 (2016)

[36] Food and Agriculture Organization of the United Nations: What factors shape small-scale farmers' and firms' adoption of new technologies? FAO, Rome (2023)

[37] Kgopa, A.T., Monchusi, B.B.: Perceptions of South African small-scale farmers towards the adoption of smart irrigation technologies. S. Afr. J. Inf. Manag. **27**(1), a1962 (2025)

[38] Marshal, R.: Use of drones in agriculture: a bird deterrent system. Undergraduate thesis, Ashesi University (2022)

[39] Manson, A.D., et al.: Avian pest damage and mitigation in African agriculture: a review. Afr. J. Ecol. **60**(2), 283–295 (2022)