

## Article

# Swarm Drones with QR Code Formation for Real-Time Vehicle Detection and Fusion Using Unreal Engine

Alaa H. Ahmed <sup>1,2,3,\*</sup>  and Henrietta Tomán <sup>1</sup> 

<sup>1</sup> Department of Data Science and Visualization, Faculty of Informatics, University of Debrecen, H-4032 Debrecen, Hungary; toman.henrietta@inf.unideb.hu

<sup>2</sup> Doctoral School of Informatics, University of Debrecen, H-4032 Debrecen, Hungary

<sup>3</sup> Department of Information Technology, College of Computer Science and Information Technology, University of Kirkuk, Kirkuk 30061, Iraq

\* Correspondence: alaa.ahmed@inf.unideb.hu

## Abstract

A single drone collects data, but a fleet builds a complete picture, and this is the primary objective of this study. To address this goal, a swarm-based drone system has been designed in which multiple drones follow one another to collect data from diverse perspectives. Such a strategy demonstrates strong potential for use in critical fields such as search and rescue operations. This study introduces the first unified framework that integrates autonomous formation control, real-time object detection, and multi-source data fusion within a single operational UAV-swarm system. A high-fidelity simulation environment was built using Unreal Engine with the AirSim plugin, featuring a lightweight QR code tracking algorithm for inter-drone coordination. The drones were employed to detect vehicles from various angles in real time. Two types of experiments were conducted: the first used a pretrained YOLO model, and the second used a custom-trained YOLOv8-nano model, which outperformed the baseline by achieving an average detection confidence of 90%. Finally, the results from multiple drones were fused using various techniques including temporal, probabilistic, and geometric fusion methods to produce more reliable and robust detection results.

**Keywords:** swarm drones; YOLO; object detection; real time; fusion techniques; Unreal Engine



Academic Editor: Simon Tomažič

Received: 17 October 2025

Revised: 29 November 2025

Accepted: 1 December 2025

Published: 3 December 2025

**Citation:** Ahmed, A.H.; Tomán, H. Swarm Drones with QR Code Formation for Real-Time Vehicle Detection and Fusion Using Unreal Engine. *Automation* **2025**, *6*, 87. <https://doi.org/10.3390/automation6040087>

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Drones, also known as Unmanned Aerial Vehicles (UAVs), are a type of aircraft that can fly without a pilot. They can be controlled remotely using radio waves or autonomously by determining their flight path in advance [1]. Drones come in many sizes and shapes and can be equipped with additional devices such as cameras or sensors for monitoring purposes. The use of drones expanded in civilian life; people begun employing them in many areas such as filming, photography, surveying, searching and rescuing. Nowadays, UAVs also have many commercial applications such as delivering packages for Amazon [2]. Even in agriculture, they have been used to manage farm tasks, monitor crops, and assist in irrigation. Moreover, drones play an essential role in improving efficiency in certain companies, especially in warehouses. They automatically detect objects' barcodes, reducing human effort and energy consumption [3]. To enhance drones' efficiency, a range of advanced techniques in image processing, object tracking, and data fusion are used.

In addition, employing a drone swarm is more efficient than just operating an individual drone in specific tasks [4]. Over the past few years, many researchers have focused on drone formation strategies to ensure both safety and efficiency [5]. Balsam Alkouz et al. [6] showed that a swarm of drones is more efficient in delivering packages than a single drone, particularly when multiple packages need to be delivered simultaneously. Petr Skobelev et al. [7] used a swarm of drones in agriculture to survey larger areas of farmland. They used smart programs to make each drone aware of its tasks and destinations. The collaborative work allowed the swarms to complete the survey more quickly and efficiently than a single drone. Carlos Carbone et al. [8] also employed drone swarms for agricultural tasks to enhance efficiency and accuracy in field monitoring. Instead of surveying the entire field, the drones focused on areas with high information gain, using this as a key criterion to prioritize the most informative regions. This strategy concentrated efforts only on regions where valuable data could be gathered.

Although previous studies have individually addressed object detection, drone formation, and data fusion, to the best of our knowledge, this research is the first to integrate all these components into a unified and comprehensive swarm-based framework. The proposed system is structured in three key phases:

1. Implementation of multi-drone formation control strategy, where drones follow each other using a QR code.
2. Deployment of onboard cameras coupled with YOLOv8 for real-time vehicle detection using two types of distinct experiments.
3. The detection results were fused using a combination of temporal, probabilistic, and geometric fusion methods to obtain more reliable and accurate outputs.

The structure of this research paper is organized as follows. Section 2 presents a review of related work and recent studies on drones. Section 3 provides a detailed description of the materials and methods used in this study, including the software and platforms. Section 4 elaborates on the detection techniques, as well as the implemented fusion methodologies. Section 5 discusses the experimental setup and the evaluation of the proposed approach. Finally, Section 6 presents the conclusion, summarizing the key findings and contributions of the research. Throughout the manuscript, the terms “UAV”, “drone”, and “drone swarm” all refer to the same type of multirotor aerial vehicle.

## 2. Related Works

In recent years, drone-related research has gained significant interest in the scientific community. Extensive research has been conducted to explore various aspects of Unmanned Aerial Vehicles. In this paper, we focus on studies that concentrate on image extraction and object detection methodologies using UAVs. Kalinov et al. [3] investigated the use of UAVs in warehouses for autonomous warehouse stocktaking. Since warehouses have challenging conditions such as poor lighting and the absence of GPS in indoor environments, they used 1D barcodes as visual landmarks for UAV localization. Each drone was equipped with a camera and a CNN barcode detector to detect barcodes and adjust their path in real time. Their real-world experiments showed that active barcode anchoring enhanced positioning accuracy by approximately 38%. Hyeon et al. [9] used a drone for continuous detection of 2D barcodes; in case the drone did not find any barcode, it started to search the entire region. They used LBP and HOG methods to detect the candidate place along with the SVM classification method. Li-Yu et al. [10] studied dynamic object tracking autonomously using drones. They utilized YOLOv4-Tiny model for detecting the required object and then applied Kalman Filter with 3D object pose estimation to enhance the interpretation. Ancy Micheal et al. [11] proposed a novel method to detect and track targets using UAVs. They focused on using a Deeply Supervised Object Detector to detect the required object and

then used a Long Short-Term Memory network for tracking. Their model achieved a lower average time compared to other models, along with 96.13% precision and 95.28% recall. Seoungjun lee et al. [12] studied rescuing people from natural disasters using UAVs. They equipped drones with practical hardware and modern sensors to make rescue operations easier and faster, especially in narrow places. They used the DJI Matrice 100 equipped with Hokuyo LiDAR sensor for global mapping, along with an Intel RealSense sensor for local mapping. Then, by fusing those sensors, they could obtain high-accuracy results which helped in finding and rescuing people.

In addition, numerous investigations have emphasized the advantages of using drone swarms in handling complex tasks, highlighting improved efficiency and productivity compared to a single drone [13]. A swarm refers to a set of drones that collaborate and coordinate to achieve the required goal [14,15].

Swarm has the ability to solve complex tasks and eliminate many single-drone limitations [16]. Jimmy Chiun et al. [17] created a system to make a swarm of small drones, called Craziflies, fly together as a team. To achieve accurate localization and orientation, they used AprilTags in the environment. Each drone was equipped with cameras to recognize those Tags, which helped them to determine their position and direction. Thus, their operation did not depend on expensive navigation systems like GPS; instead they used AprilTags for localization. In 2023, Moon et al. [18] developed a swarm-based drone system to detect objects in real time. They equipped each drone with a camera and lightweight YOLOv5 to detect objects such as a missing person or specific item during flight. The results were streamed to the Ground Control Station for monitoring large areas in critical missions.

Previous work primarily focused on formation control or reaching the desired location but has rarely addressed the integration of swarm coordination with high-level tasks such as object detection in real time. In contrast, we used a QR code-based formation system to make drones follow each other and maintain stable formation. At the same time, those drones work together to detect vehicles in real time. By fusing their outputs, the system achieves higher accuracy and greater reliability. This fusion-based multi drone system is suitable for many tasks such as real-time surveillance and monitoring applications. Moreover, to facilitate the training of the drone system, a realistic environment was constructed using Unreal Engine with AirSim plugin.

### 3. Materials and Methods

#### 3.1. Unreal Engine

Unreal Engine (UE) is one of the most famous simulation platforms developed by Epic Games. It can create 2D and 3D simulations providing a realistic virtual environment. It is a development tool used mostly in game designs, broadcast, film production, and any real-time programs [19]. UE represents a huge revolution in game development industries since it provides platforms that allow creators to develop high-quality games. It offers highly realistic graphics, cinematic lighting effects, and advanced simulations.

#### 3.2. Microsoft AirSim

Aerial Informatics and Robotics Simulation (AirSim) is a powerful simulation framework built on Unreal Engine [20]. It provides physically and visually realistic simulations, offering a cost-effective and time-efficient alternative to real-world testing [21]. Moreover, it delivers high-fidelity rendering and supports GPS, LiDAR, and IMU sensors [22]. Unreal Engine and AirSim are connected via their APIs, with Python 3.9 serving as the primary language for implementation and interaction.



The reason for using QR digital transfer rather than visual scanning is its superior speed, robustness, and reliability. It requires only a few milliseconds to encode and transmit the data, which is significantly faster than detecting and decoding visual codes using cameras. Also, visual scanning is highly affected by environmental factors such as distance, lighting conditions, and camera resolution. In contrast, the digital QR protocol overcomes those dependencies by encoding the data to Base64 and transmitting it using UDP, regardless of environmental conditions. Additionally, QR code provides a built-in error-correction mechanism, which allows reliable decoding even under partial data loss or transmission noise, outperforming barcodes and AprilTags.

#### 4.3. Drone Formation Strategy

All drones fly at the same altitude, with the follower drones positioned at a fixed distance of 10 m behind the lead drone. To prevent collisions, the follower drones are slightly offset along the lateral axis, as shown in Figure 3. So, when the lead drone starts to take off and move forward, the follower drones simultaneously take off and adjust their position based on the current location of the lead drone. In case any follower drone comes closer to the lead drone than predefined safety threshold (e.g., 5 m), it will stop to prevent collision. All the drones are engaged in a chained configuration, such that each drone computes its reference waypoint relative to the vehicle immediately ahead as shown in Figure 3.

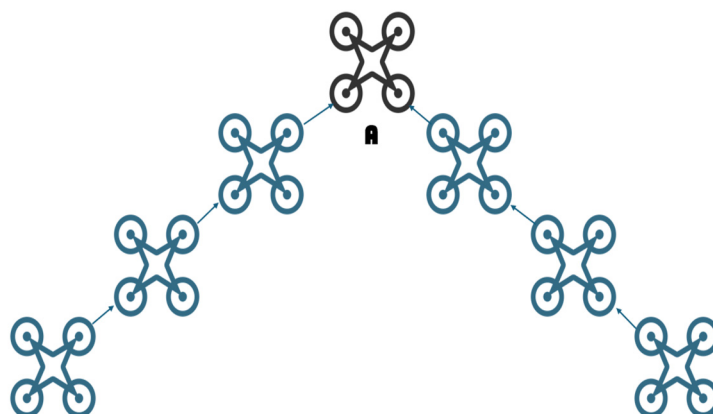


**Figure 3.** Swarm of drones is led by drone A.

To control the movement of the first drone, we used two approaches:

1. Automated Control: The software manages the drone's movement autonomously.
2. Manual Control: A human operator controls the drone's movement directly.

During formation, each follower drone needs to estimate the location of the lead drone, which is determined by a central coordinator ( $x_{val}$ ,  $y_{val}$ ,  $z_{val}$ ). Meanwhile, the position of each non-leading drone ( $x$ ,  $y$ ,  $z$ ) is determined depending on the position of the preceding drone represented by the coordinates ( $x_{prev}$ ,  $y_{prev}$ ,  $z_{prev}$ ) using the following logic.

$$x = x_{prev} - fd,$$

$$y = y_{prev} + y_{offset},$$

$$z = z_{prev} = alt,$$

where  $fd$  represents the desired following distance,  $y_{offset}$  denotes the lateral displacement, and  $alt$  is the altitude used by all drones.

At each step, the system calculates the distance between the drones to decide whether to move or pause to keep them safe. After determining the locations, each non-leading

drone starts moving and follows the previous one automatically. The reason for choosing the leader–follower formation is that we aim to collect multi-view observations of the same target using a drone swarm. Using a chained leader–follower formation is more practical and beneficial in achieving the required goal while keeping the system simple, safe, and lightweight. It reduces collision risks and coordination complexity since each drone relies only on the nearest neighbor, not the entire swarm. In other words, this formation provides multi-drone coverage, low coordination complexity, and compact footprint; it is very suitable for vehicle detection tasks more than complex topologies such as parallel or circular, which require additional synchronization, more competition, and larger space.

#### 4.4. Vehicle Detection

Vehicle detection aims to analyze data captured by cameras or sensors and extract relevant objects such as cars and trucks. In this study, YOLOv8 was employed as the main detection framework due to its high accuracy and real-time inference capability. The architecture of YOLOv8 partitions each input image into grid and predicts bounding boxes, class probability, and confidence score, accepting only those above the defined threshold. The model is pretrained using COCO, which is a large-scale dataset consisting of 200,000 labeled images across 80 various object categories such as vehicles and animals. Pretrained model allows YOLOv8 to recognize vehicle types effectively in real time.

To further evaluate the performance of the object detection pipeline under realistic conditions, a Synthetic Drone Dataset was generated using Unreal Engine. The generated dataset includes two object categories, cars and trucks, and comprises 500 images captured using a virtual RGB camera mounted on a drone. The images are taken from different viewpoints, lighting conditions, distances, and vehicle poses. This shows that the dataset is diverse enough to train a model without overfitting problems. Each image has a resolution of  $(666 \times 333)$  pixels. To annotate those images efficiently, an annotations platform called Computer Vision Annotation Tool (CVAT) was utilized by drawing a bounding box around each visible vehicle and labeling them accordingly. Since YOLOv8 requires annotations in a specific format, the dataset was reformatted accordingly before training process. All the images were scaled to  $640 \times 640$  pixels to preserve consistency in model training. Figure 4 shows the result of three drones detecting vehicles with various confidence scores using the COCO dataset and Synthetic Drone Dataset.



**Figure 4.** Three drones detecting vehicles with different confidence scores (a) using custom-trained model and (b) pretrained model.

#### 4.5. Fusion Methodology

Data Fusion is a specialized form of data integration that merges the output from multiple sources to produce more comprehensive results. Unlike standard integration, fusion focuses on enhancing and extracting only the required data without redundancy. In

this study, we used fusion techniques to obtain the most useful observation from the drones. Once vehicle detections were obtained, the outputs from individual drones were fused to achieve accurate and reliable results. The system computes the fused position of the detected object by integrating the observations from the drones. To obtain efficient results in terms of accuracy and robustness, various types of fusion techniques were applied: temporal fusion methods such as Kalman Filter, Extended Kalman Filter, and Unscented Kalman; Probabilistic fusion as Bayesian filter; and geometric fusion as IoU-based fusion, as shown in Figure 5.

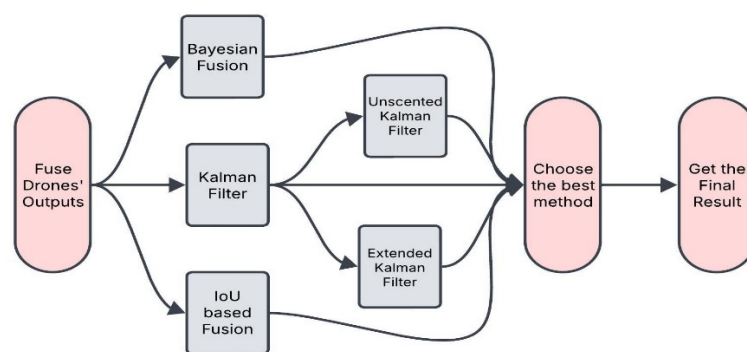


**Figure 5.** Flow diagram of fusion techniques.

First, we used Kalman Filter (KF), which is one of the best fusion techniques that works mostly with linear systems [23]. It combines the noisy data from the drones and produces a single consistent output. Each drone provides detection in the form of (x-center, y-center, height, width). The process of KF involves two main steps: prediction and update. In the prediction step, it estimates the next bounding box of the object depending on its current state. In the update step, it receives new measurements (detections) and compares them with the prediction ones to produce more accurate estimations. It calculates the overlap between the detection and prediction bounding boxes and returns 1 if the overlap exceeds the threshold, and 0 otherwise:

$$\text{IoU} = \frac{\text{Area of intersection}}{\text{Area of union}} \quad (1)$$

KF uses these results to merge detections and assembles the final fused boundary box by smoothing them using Kalman filters.

Second, we used Extended Kalman Filter (EKF), which is one of the subsets of KF but is primarily used with nonlinear systems. It can deal with more complex motions. This method has the same steps as the KF, but Jacobian matrix is used for linearization in case the system is nonlinear. This technique also helps in removing redundant detections and combining them into a single accurate set of detections.

Third, Unscented Kalman Filter (UKF) was employed, which is an advanced subset of KF designed for nonlinear system. It follows similar approaches as EKF, but instead of the Jacobian matrix, sigma points are used to deal with nonlinear systems.

Fourth, we used IoU-based fusion which combines redundant detections while maintaining high accuracy. The algorithm calculates the overlap between the boundary boxes as in Equation (1). It checks if the distance between the centers of the detections is near enough to prevent merging incorrect objects. The distance between the centers can be calculated using Euclidian distance as

$$\text{Center}_{\text{Dist}} = \sqrt{(X_{c1} - X_{c2})^2 + (Y_{c1} - Y_{c2})^2} \quad (2)$$

If the centers are so close, then it merges them considering them belonging to the same object. Then, it uses a weighted average to refine the boundary box coordinates. This ensures that the bounding boxes of the same object are merged consistently and without any discrepancies.

Lastly, we used Bayesian Fusion (BF), which is very effective in fusing multi-sensors by combining their measurements to handle noisy detections [24]. The algorithm iterates through all the detections to check the overlap between them. In contrast to other algorithms, Bayesian Fusion uses the confidence score as weights, so that the detections with higher confidence scores have a strong influence. Then, to refine the bounding box over multiple detections, it uses a Kalman-like approach.

The reason for choosing these types of fusion techniques is that the temporal method, which includes Kalman filter variants, is well suited for the follower–leader motions, which are smooth, mostly linear, and low acceleration. These characteristics make temporal methods more efficient and appropriate for the scenario. EKF and UKF are used to account for nonlinearity arising from formation offsets and coordinate transformations. Probabilistic fusion, which includes the Bayesian approach, combines confidence scores from multiple drones and works perfectly, especially when multi-view detections have varied levels of reliability. The geometric fusion, which includes IoU-based fusion, aggregates overlapping bounding boxes and eliminates redundancy, providing an efficient measurement-level fusion mechanism for multi-view detections.

## 5. Results and Evaluation Metrics

### 5.1. Simulation Setup

The simulation was performed on a computer that runs on a 13th Gen Intel(R) Core (TM) i7-13620H 2.40 GHz processor with 16.0 GB of memory and an NVIDIA GEFORCE RTX 4050 6 GB GPU. For simulation, we used Unreal Engine, which interacted with AirSim APIs using Python programming language.

### 5.2. Communication Evaluation

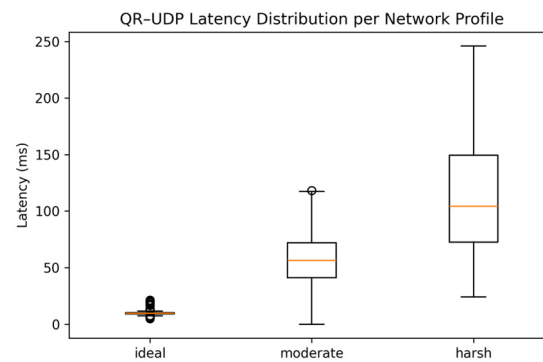
To evaluate the robustness of the proposed QR code-based UDP transmission protocol, the simulation was integrated with a configurable network impairment model which includes three profiles: Ideal, Moderate, and Harsh. Each profile was tested with 400 transmissions, and several communication metrics were measured. To emulate realistic wireless degradations, packet loss, latency, error-bit corruption, and bandwidth limits were injected:

1. Sent rate is a fraction of the successful transmitted packets to the total number of packets attempted.
2. Decode success is a fraction of successful message reconstruction (Follow\_Leader\_1) to the total number of trails.
3. Corruption rate is a fraction of corrupted packets, where bits are flipped by the BER model, relative to the total number of sent packets.
4. Latency is the time it takes from the moment the packet is sent until the followers receive and decode it. Both mean and 95th percentile latency was considered.

The Ideal network experienced 0% packet loss with approximately 10 ms mean latency and unlimited bandwidth, as shown in Table 1. The Moderate network exhibited 5% packet loss with 58 ms mean latency and 1 Mbps bandwidth. Under the Harsh conditions, the network showed 20% packet loss, 111 ms mean latency, and 256 kbps bandwidth. In other words, the QR-UDP remained highly reliable under aggressive conditions of packet loss and bit-error, maintaining a decode success close to 80% under the Harsh conditions, as shown in Figure 6.

**Table 1.** The communication performance under each profile.

| Profile  | Sent Rate | Decode Success | Corruption Rate | Mean Latency (s) | 95th% Latency (s) |
|----------|-----------|----------------|-----------------|------------------|-------------------|
| Ideal    | 1.000     | 1.000          | 0.0000          | 0.0100           | 0.0140            |
| Moderate | 0.940     | 0.938          | 0.0025          | 0.0582           | 0.0885            |
| Harsh    | 0.818     | 0.795          | 0.0225          | 0.1105           | 0.1979            |

**Figure 6.** The latency distribution for each network profile.

### 5.3. Formation Evaluation

To evaluate how accurate the follower drones maintain their formation using QR-based location updates, we used the same three network profiles: Ideal, Moderate, and Harsh as is shown in Table 2. Three formation metrics were computed:

1. Mean error is the average difference between the correct position and the estimated one. A lower mean error means the system is more accurate.
2. Standard deviation (std) is the consistency of the formation error. A lower std value indicates more stable and predictable behaviors.
3. Max error is the worst-case deviation observed during the experiments. It shows the extreme mistakes that follower drones make. A lower max error is better for avoiding collision and keeping the format without losing the track.

**Table 2.** Summarizes the formation accuracy for each network profile.

| Profile  | Mean Error | Std   | Max Error |
|----------|------------|-------|-----------|
| Ideal    | 0.798      | 0.368 | 1.861     |
| Moderate | 0.906      | 0.950 | 8.645     |
| Harsh    | 1.061      | 1.084 | 8.589     |

The leader drone follows a straight reference, while the follower drones maintain a fixed offset, as described earlier in Section 4. At each control step, the lead drone transmits the formation command to the follower drones. Then, the follower drones update their locations depending on the valid QR command. If the QR code is successfully decoded, they are in the right position; otherwise they continue to track their last valid QR command. To compute the accuracy, we recorded the Euclidean distance between each follower and their correct formation position. The results indicate a gradual degradation of performance such that the mean formation error increases 0.79 m in the Ideal case to 0.90 m in the Moderate case, and 1.06 in the Harsh case, demonstrating stability and predictability behavior even under tough conditions, as shown in Figure 7.

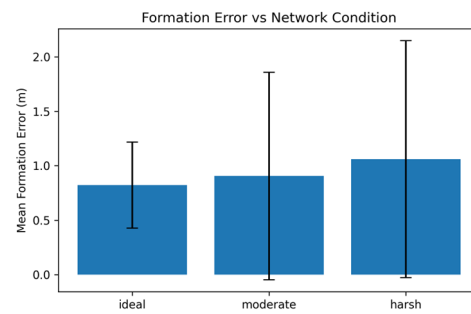


**Figure 7.** Formation error distribution per network profile.

#### 5.4. Experiments Using Synthetic Drone Dataset

In this study, we trained a YOLOv8-nano model for vehicle detection on a Synthetic Drone Dataset. The trained model demonstrated a remarkable ability to identify various vehicle types with a high average-confidence score exceeding 90%, outperforming the COCO-trained baseline under our evaluation setup.

To assess the performance and reliability of the trained model, we utilized precision and F1 score as the primary metrics. Precision measures the proportion of correctly identified vehicles among the total detections made by the model, providing a strong indicator of its accuracy and effectiveness in distinguishing between different vehicle types. Additionally, we used F1 score, which is the harmonic mean of precision and recall for a specific class. The precision and F1 score results are presented in Figure 8.

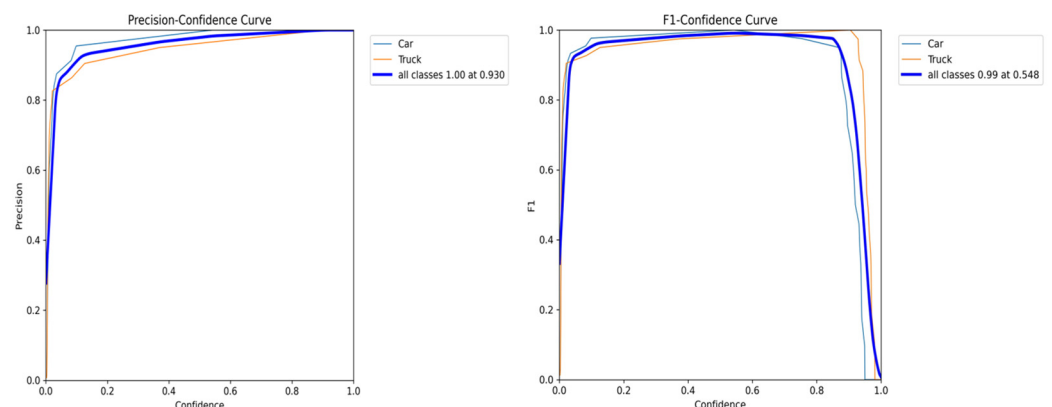


**Figure 8.** Precision and F1–confidence curves of the custom-trained model.

The precision–confidence curve shows that once the confidence threshold exceeds 0.2, the model performs remarkably well with very few false positives. When the threshold reaches 0.9, as shown in Table 3, the model achieves perfect precision for all classes. In simpler terms, it arrives at a point of complete accuracy, almost without any mistakes in vehicle detection.

**Table 3.** Vehicle detection results using pretrained and custom-trained models.

| Drone ID | Using COCO Model                   |                                      | Using Our Trained Model            |                                      |
|----------|------------------------------------|--------------------------------------|------------------------------------|--------------------------------------|
|          | Confidence Score of Detecting Cars | Confidence Score of Detecting Trucks | Confidence Score of Detecting Cars | Confidence Score of Detecting Trucks |
| Drone1   | 0.56                               | 0.56                                 | 0.91                               | 0.92                                 |
| Drone2   | 0.61                               | 0.66                                 | 0.90                               | 0.91                                 |
| Drone3   | 0.79                               | 0.54                                 | 0.90                               | 0.91                                 |

Meanwhile, the F1–confidence curves for cars, trucks, and a combined curve for both classes indicate that the model maintains high precision and recall across a broad range of confidence thresholds. This demonstrates the model’s ability to detect vehicles accurately while minimizing both false positives and false negatives. The model achieves its best performance when F1 score is near 0.99 at a confidence threshold of 0.55.

### 5.5. Detection Evaluation

The drones detect vehicles directly in real time, as soon as they take off. The average confidence scores for detecting vehicles by the drones using the COCO model and the custom-trained model show that our trained model achieves higher results than the COCO model. The average confidence score using the COCO dataset reaches a maximum of 0.79, while custom-trained model exceeds 90%, as shown in Table 3.

### 5.6. Fusion Evaluation

Fusion strategies were assessed utilizing precision as the primary metric due to the absence of per-frame ground-truth alignment in the merged dataset, which makes recall unreliable. Precision can be defined as

$$\text{Precision} = \frac{\sum \text{True Positive}}{\sum \text{True Positive} + \sum \text{False Positive}} \quad (3)$$

Table 4 illustrates precision outcomes of fusion strategies, offering a credible metric for detection reliability and demonstrating the efficacy of the fusion process in minimizing false detections across multi-drone views. The evaluation uses two Intersection over Union thresholds, 0.3 and 0.5, to assess performance under different conditions. KF, UKF, and BF achieve higher precision in both models, particularly on the synthetic dataset. Meanwhile, EKF and IoU-based fusion show lower precision with the COCO-based model but achieve optimal results with the custom-trained model. The comparable behaviors of the Kalman Filters (KF, EKF, and UKF) arise from system’s approximately linear dynamics, leading the three methods to yield nearly equivalent estimations.

**Table 4.** Precision results of fusion techniques using pretrained and custom-trained models.

| Fusion Techniques       | Precision Using COCO Dataset |               | Precision Using Synthetic Dataset |               |
|-------------------------|------------------------------|---------------|-----------------------------------|---------------|
|                         | Threshold 0.5                | Threshold 0.3 | Threshold 0.5                     | Threshold 0.3 |
| Kalman Filter           | 0.7143                       | 0.8000        | 0.7500                            | 0.8750        |
| Extended Kalman Filter  | 0.3571                       | 0.4444        | 0.7500                            | 0.8750        |
| Unscented Kalman Filter | 0.6154                       | 0.7273        | 0.7500                            | 0.8750        |
| Bayesian Fusion         | 0.6429                       | 0.7778        | 0.800                             | 0.7500        |
| IoU-Based Fusion        | 0.3448                       | 0.4091        | 0.800                             | 0.6250        |

## 6. Conclusions

We developed a high-fidelity environment using Unreal Engine to train a swarm of drones maintaining a stable formation. These drones were employed to detect vehicles from various viewpoints in real-time. To evaluate our methodology, we conducted two sets of experiments: the first one uses a pretrained YOLOv8 model with the COCO dataset, and the second one uses a custom-trained model based on a Synthetic Drone Dataset. The results demonstrated that the custom-trained model significantly outperformed the COCO-based model in terms of average confidence score—it exceeded 90% in detecting vehicles.

After the detection process, we implemented various fusion techniques to combine and enhance the observations from multiple drones. The techniques included Kalman filter, Extended Kalman Filter, Unscented Kalman Filter, Bayesian Fusion, and IoU-based fusion. The necessities of applying fusion techniques are to reduce noise, smooth out the variation in confidence score, and obtain a unified result. The evaluation of fusion techniques using precision as the primary metric demonstrate significantly better results using the Synthetic Drone Dataset compared to the COCO dataset. Several techniques achieved perfect precision score across most of the tested scenarios, which shows the robustness of our custom-trained model; specifically, KF showed the highest robust performance under all conditions with 0.87 precision using the custom-trained model. Overall, our study highlights the effectiveness and critical role of fusion techniques in optimizing the detection results using multi-drone systems. This research represents the first phase of a larger, ongoing work. In this phase, we mainly focused on communication and the formation of drone swarm in a static environment so the proposed system could be evaluated in a reproducible and controlled environment. Future work will build on this foundation by handling more complex environments with static and dynamic obstacles, using obstacle avoidance techniques in real time, and multi-class detections. The extension parts are already under investigation and are being developed to further generalize and strengthen the proposed multi-drone framework.

**Author Contributions:** Conceptualization, A.H.A.; methodology, A.H.A.; software, A.H.A.; validation, A.H.A.; formal analysis, A.H.A.; investigation, A.H.A.; resources, A.H.A.; data curation, A.H.A.; writing—original draft preparation, A.H.A.; writing—review and editing, A.H.A. and H.T.; visualization, A.H.A.; supervision, H.T.; funding acquisition, H.T. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the University of Debrecen Program for Scientific Publication.

**Data Availability Statement:** The dataset used in this study is not publicly available.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Mohsan, S.A.H.; Khan, M.A.; Noor, F.; Ullah, I.; Alsharif, M.H. Towards the unmanned aerial vehicles (UAVs): A comprehensive review. *Drones* **2022**, *6*, 147. [\[CrossRef\]](#)
2. Bappy, A.; Asfak-Ur-Rafi, M.; Islam, M.S.; Sajjad, A.; Imran, K.N. Design and Development of Unmanned Aerial Vehicle (Drone) for Civil Applications. Ph.D. Thesis, BRAC University, Dhaka, Bangladesh, 2015.
3. Kalinov, I.; Petrovsky, A.; Ilin, V.; Pristanskiy, E.; Kurenkov, M.; Ramzhaev, V.; Idrisov, I.; Tsetserukou, D. Warevision: CNN barcode detection-based UAV trajectory optimization for autonomous warehouse stocktaking. *IEEE Robot. Autom. Lett.* **2020**, *5*, 6647–6653. [\[CrossRef\]](#)
4. Zhang, P.; Wang, Z.; Zhu, Z.; Liang, Q.; Luo, J. Enhanced multi-UAV formation control and obstacle avoidance using IAAPF-SMC. *Drones* **2024**, *8*, 514. [\[CrossRef\]](#)
5. Enwerem, C.; Baras, J.; Romero, D. Distributed optimal formation control for an uncertain multi-agent system in the plane. *arXiv* **2023**, arXiv:2301.05841.
6. Alkouz, B.; Bouguettaya, A.; Mistry, S. Swarm-based Drone-as-a-Service (SDAAS) for delivery. In Proceedings of the 2020 IEEE International Conference on Web Services (ICWS), Beijing, China, 19–23 October 2020; pp. 441–448.
7. Skobelev, P.; Budaev, D.; Gusev, N.; Voschuk, G. Designing multi-agent swarm of UAV for precise agriculture. In Proceedings of the International Conference on Practical Applications of Agents and Multi-Agent Systems, Toledo, Spain, 20–22 June 2018; pp. 47–59.
8. Carbone, C.; Albani, D.; Magistri, F.; Ognibene, D.; Stachniss, C.; Kootstra, G.; Nardi, D.; Trianni, V. Monitoring and mapping of crop fields with UAV swarms based on information gain. In Proceedings of the International Symposium on Distributed Autonomous Robotic Systems, Kyoto, Japan, 1–4 June 2021; pp. 306–319.

9. Cho, H.; Kim, D.; Park, J.; Roh, K.; Hwang, W. 2D barcode detection using images for drone-assisted inventory management. In Proceedings of the 15th International Conference on Ubiquitous Robots (UR 2018), Honolulu, HI, USA, 26–30 June 2018; pp. 461–465.
10. Lo, L.Y.; Yiu, C.H.; Tang, Y.; Yang, A.S.; Li, B.; Wen, C.Y. Dynamic object tracking on autonomous UAV system for surveillance applications. *Sensors* **2021**, *21*, 7888. [[CrossRef](#)] [[PubMed](#)]
11. Micheal, A.A.; Vani, K.; Sanjeevi, S.; Lin, C.H. Object detection and tracking with UAV data using deep learning. *J. Indian Soc. Remote Sens.* **2021**, *49*, 463–469. [[CrossRef](#)]
12. Lee, S.; Har, D.; Kum, D. Drone-assisted disaster management: Finding victims via infrared camera and LiDAR sensor fusion. In Proceedings of the 3rd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), Nadi, Fiji, 5–6 December 2016; pp. 84–89.
13. Catala-Roman, P.; Segura-Garcia, J.; Dura, E.; Navarro-Camba, E.A.; Alcaraz-Calero, J.M.; Garcia-Pineda, M. AI-based autonomous UAV swarm system for weed detection and treatment: Enhancing organic orange orchard efficiency with Agriculture 5.0. *Internet Things* **2024**, *28*, 101418. [[CrossRef](#)]
14. Campion, M.; Ranganathan, P.; Faruque, S. UAV swarm communication and control architectures: A review. *J. Unmanned Veh. Syst.* **2018**, *7*, 93–106. [[CrossRef](#)]
15. Alqudsi, Y.; Makaraci, M. UAV swarms: Research, challenges, and future directions. *J. Eng. Appl. Sci.* **2025**, *72*, 12. [[CrossRef](#)]
16. Shrit, O.; Martin, S.; Alagha, K.; Pujolle, G. A new approach to realize drone swarm using ad-hoc network. In Proceedings of the 16th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), Budva, Montenegro, 28–30 June 2017; pp. 1–5.
17. Chiun, J.; Tan, Y.R.; Cao, Y.; Tan, J.; Sartoretti, G. STAR: Swarm technology for aerial robotics research. In Proceedings of the 24th International Conference on Control, Automation and Systems (ICCAS), Jeju, Republic of Korea, 29 October–1 November 2024; pp. 141–146.
18. Moon, S.; Jeon, J.; Kim, D.; Kim, Y. Swarm reconnaissance drone system for real-time object detection over a large area. *IEEE Access* **2023**, *11*, 23505–23516. [[CrossRef](#)]
19. Jirkal, J. Drone Simulation Using Unreal Engine. 2020. Available online: <https://www.semanticscholar.org/paper/Drone-Simulation-Using-Unreal-Engine-Jirkal/969501f3291e182ceda34a5062c7c70a2ef2b130> (accessed on 12 July 2025).
20. Madaan, R.; Gyde, N.; Vemprala, S.; Brown, M.; Nagami, K.; Taubner, T.; Cristofalo, E.; Scaramuzza, D.; Schwager, M.; Kapoor, A. Airsim drone racing lab. In Proceedings of the NeurIPS 2019 Competition and Demonstration Track, Vancouver, BC, Canada, 8–14 December 2019; pp. 177–191.
21. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics: Results of the 11th International Conference*; Springer: London, UK, 2017; pp. 621–635.
22. Chao, Y.; Dillmann, R.; Roennau, A.; Xiong, Z. E-DQN-based path planning method for drones in AirSim simulator under unknown environment. *Biomimetics* **2024**, *9*, 238. [[CrossRef](#)] [[PubMed](#)]
23. Ahmed, A.H.; Toman, H. Stochastic fusion techniques for state estimation. *Computation* **2024**, *12*, 209. [[CrossRef](#)]
24. Ahmed, A.H.; Sadri, F. Datafusion: Taking source confidence into account. In Proceedings of the 8th International Conference on Information Systems and Technologies (ICIST), Istanbul, Turkey, 16–18 March 2018; pp. 1–6.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.