BS-CYOLOv5: A Deep Learning and Backtracking Search Framework for Multi-Sensor UAV Navigation and Obstacle Avoidance

Zongqiang Deng
China People's Police University, Basic Courses Teaching Department of Guangzhou Campus, Guangzhou, Guangdong, 510663, China
E-mail: jenekop2006@126.com

Keywords: unmanned aerial vehicles (UAV), autonomous navigation, obstacle avoidance, backtracking search-mutated customized YOLOv5 (BS-CYOLOv5), Deep Learning (DL)

Recieved: May 13, 2025

Unmanned Aerial Vehicles (UAVs) are increasingly used in applications such as aerial surveillance, disaster response, and urban air mobility. Autonomous navigation and obstacle avoidance are critical for UAVs to operate safely in dynamic environments. Traditional obstacle avoidance methods rely on predefined rules and sensor-based heuristics, which struggle with real-time adaptability in complex environments. This presents a Deep Learning (DL)-based Backtracking Search-mutated Customized YOLOv5 (BS-CYOLOv5) framework, integrating multi-sensor data processing, real-time obstacle detection, and optimized path planning. The dataset includes 10,000 labeled UAV flight data samples recorded using RGB cameras, LiDAR, IMU, and GPS, the gathered data was split with 80% training and 20 % testing. Data pre-processing techniques, such as normalization, are applied to enhance model performance and generalization. For obstacle detection, the CYOLOv5 model is employed due to its high detection accuracy, enabling real-time identification of obstacles in various environments. Navigation and path planning are optimized using the Backtracking Search Algorithm (BSA), which dynamically adjusts flight trajectories to ensure efficient and collision-free navigation. Although these components play independent functions, they are tightly integrated into the BS-CYOLOv5 architecture, where BS also fine-tunes detection-related hyper parameters to improve performance. Experimental results demonstrate that the proposed approach enhances UAV obstacle avoidance accuracy and navigation efficiency. The results reveal statistically substantial improvements in the obstacle detection accuracy (98%) and high precision, recall, f1-score and IOU, when evaluated on the full 2,000-image test set. It contributes to advancing intelligent UAV systems by integrating state-of-the-art DL and optimization techniques for enhanced autonomy and safety in real-world scenarios.

Povzetek: Članek obravnava samodejno navigacijo UAV-jev, kjer klasične metode slabo delujejo v dinamičnih okoljih. Predlaga BS-CYOLOv5, ki združuje večsenzorsko zaznavanje z izboljšanim YOLOv5 in Backtracking Search za optimizacijo hiperparametrov ter poti.

1 Introduction

The widespread popularity of UAVs, also known as drones, results from their multiple military and commercial uses and research applications [1]. UAV efficiency alongside safety and environment adaptation strongly depends on reliable autonomous navigation functionality combined with obstacle avoidance capabilities. The autonomous navigation systems of UAVs make them capable of executing surveillance tasks as well as search and rescue operations together with environmental monitoring and logistics without needing continuous human supervision [2]. UAVs benefit from this function, which eliminates operator overreach while improving operational output and decreasing the chance of mishaps. The obstacle avoidance feature enables UAVs to safely traverse complicated spaces without accidents, thus improving their mission-performance accuracy [3]. UAV platforms have advanced in terms of mobility, selectivity, accuracy, short operation cycles, low maintenance costs, practicality, and safety [4].

UAV usage continues to expand within agricultural operations and delivery service sectors alongside infrastructure assessment and disaster relief responsibilities, which underlines the standards for improved guidance systems and obstacle detection capabilities [5]. Drones may modify their flying routes in response to environmental circumstances and unanticipated impediments because of autonomous navigation, which improves UAV decision-making and reactivity. An efficient obstacle detection system decreases the possibility of UAV or surrounding infrastructure damage for better operational reliability and mission accomplishment [6]. Figure 1 displays the obstacle avoidance and autonomous navigation technologies for UAVs.

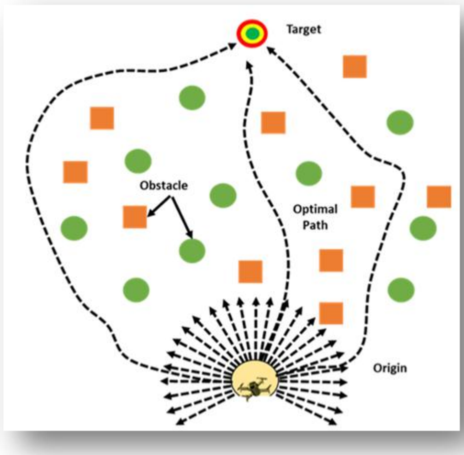

Figure 1: Obstacle avoidance and autonomous navigation technologies for UAV

Technological developments for UAV systems will concentrate on developing precise navigational capabilities and dependable detection mechanisms as well as enhancing power efficiency and response speeds over time [7]. The safe and effective operation of UAVs depends on self-navigational systems combined with obstacle detection capabilities in different operational settings. Continuous research into UAV technology has improved its capabilities, opening up new business prospects and enabling risky, complex operations [8]. Performance stability remains a challenge for diverse, unpredictable conditions, although significant progress has been made. The performance of sensors and navigation precision becomes compromised by environmental factors, which include wind conditions combined with rain and changing light illumination. This investigation presents a DL-based BS-CYOLOv5 framework integrating multi-sensor data processing, real-time obstacle detection, and optimized path planning. The research investigates the integration of deep learning and optimization approaches to improve UAV autonomous navigation. The following questions direct the research's focus towards detection accuracy, model optimization, and real-world adaptability.

1. How can deep learning models be tuned to improve real-time obstacle identification accuracy in UAV navigation based on multi-sensor data?
2. How does backtracking search-based hyperparameter tuning improve YOLOv5's performance for UAV obstacle avoidance?

In Section 2, a list of relevant works was presented. Section 3 includes the methodology is described. Section 4 contains the findings of the research. The discussion portion is provided in Section 5, and Section 6 contains the conclusion.

2 Related work

To enable navigation in locations with a lot of obstacles, [9] suggested a reactive limited navigation strategy for a UAV with integrated obstacle avoidance. Nonlinear Model Predictive Control (NMPC) was the foundation of the suggested navigation architecture. The NMPC's parameter restricted the UAV's position area. A unique method for a tiny UAV's autonomous navigation in tree plantations using just one camera was examined in [10]. An ML model called the Faster Region-based Convolutional Neural Network (Faster R-CNN) was developed for the purpose of detecting tree trunks because monocular vision was unable to give depth information.

The safety issue of accidents with non-stationary objects was demonstrated in [11], which was frequently ignored because there was a dearth of technology and strategies to deal with it. The computationally demanding challenge of utilizing commercial sensors was addressed in a unique way that uses DL algorithms. Using image input, [12] investigated the soft-actor-critic technique to teach a drone to autonomously avoid obstacles in continuous action space. The findings demonstrated that by just entering the depth map, the method allowed the UAV to ignore obstacles in the testing area. Additionally, without retraining, it exhibited a greater avoidance of obstacles rate in the modified environment.

To increase productivity and assurance of drone flight safety, [13] suggested employing UAVs to perform duties like path planning and obstacle avoidance. A popular reinforcement learning state–action–reward–state–action (SARSA) algorithm was compared with the suggested method. Deep Reinforcement Learning (DRL) and probabilistic algorithms were proposed in [14] to prevent collisions that consume less energy. The suggested algorithms could be executed at the Multi-Access Edge Computing (MEC) or on top of the UAV, depending on the task overhead and UAV capacity. The suggested solutions have been tested in a challenging setting with several UAVs moving randomly and uncorrelated in a restricted region.

An actual three-dimensional (3D) path planner was proposed in [15] to guide the UAV toward its goal through an obstacle-free route. The suggested path planner was heuristic in the sense of the A⋆ method, but unlike A⋆, it does not need border nodes to be maintained in memory. The planner calculated collision-free routes by using the relative positions of identified objects, or barriers. The Dual Experience Attention Convolution Soft Actor-Critic (DAC-SAC) technique was proposed in [16]. The technique used a convolutional layer in conjunction with a self-attention method, a dual memory buffer pool, and the soft-actor-critic technique.

The UAV flight path prediction in dynamic situations was improved by utilizing Artificial Neural Networks (ANNs) [17]. A Python-based multi-hidden-layer ANN model was trained on two datasets using tenfold cross-validation. The results reveal high accuracy in distance prediction (R^2 = 99.45%, MAE = 0.158), but poor altitude prediction (R^2 = 53.95%). The method's limitation lies in its reduced performance for altitude estimation. Table 1 shows the comparative analysis of the related work.

Table 1: Comparative analysis of the related work

Ref.	Year	Area Focused	Algorithms	Limitations	Performance
[9]	2021	UAV Navigation	NMPC, Proximal Averaged Newton for Optimal Control (PANOC), Optimization Engine (OpEn), Penalty Method	Computational limitations on UAV onboard processing power	Fast solutions with limited computational power, real-time obstacle avoidance, and improved closed-loop performance
[10]	2021	UAV Navigation in Precision Agriculture	Faster Region-based CNN (Faster R-CNN), Monocular Vision	Lack of depth information from monocular vision, limited by camera accuracy in low-light conditions	Successful in 11 flight tests across two different environments, accurate and robust obstacle avoidance
[11]	2021	UAV Collision Avoidance	Deep Learning, Transfer Learning	The computational intensity of real-time processing, limited by commercial vision sensor capability	Successful real-time collision avoidance with dynamic objects, positive transfer learning results, and effectiveness in real-world tests.
[12]	2021	Drone Obstacle Avoidance	Soft-Actor-Critic (SAC)	Discrete action spaces lead to imprecise and unnatural drone movements	Improved obstacle avoidance rate in training and reconfigured environments without retraining
[13]	2023	Planning UAV Routes and Avoiding Obstacles	Q-Learning, SARSA	Energy consumption and flight duration limitations for completing cage detection	Improved path planning efficiency, effective obstacle avoidance, and enhanced flight safety in a virtual environment (AirSim)
[14]	2021	UAV Collision Avoidance and Energy Efficiency	Probabilistic and Deep Reinforcement Learning (DRL)-based algorithms	Computational limitations on UAV onboard processing power	Efficient collision avoidance, effective in familiar and unfamiliar environments, reduced energy consumption.
[15]	2021	UAV Navigation	Heuristic A*-based 3D path planner (without storing frontier nodes)	Limited computational resources on UAV onboard processing	Fast guidance, real-time obstacle avoidance, efficient in constrained environments (validated through SITL simulations and real flight tests)
[16]	2024	UAV Obstacle Avoidance	Dual Experience Attention Convolution Soft Actor-Critic (DAC-SAC), Dual Experience Buffer Pool, CNN, and Self-Attention Mechanism	Scarcity of successful training data. SAC's limitations in handling image data	84.8% success rate in an unknown environment and 99.5% success rate in a known environment.
[17]	2024	UAV Flight Path Prediction	Artificial Neural Networks (ANNs), Python-based model	Reduced accuracy in altitude prediction	R^2 = 99.45%, MAE = 0.158 for distance; R^2 = 53.95%, MAE = 15.2 for altitude

Existing approaches, such as YOLOv5, struggle with occlusions and small object recognition [16], whereas Soft Actor-Critic (SAC) performs imprecisely in dynamic UAV contexts [11]. Deep reinforcement learning algorithms have significant processing costs and slow convergence [13]. These constraints underline the importance of the proposed BS-CYOLOv5, which combines YOLOv5's quick detection with adaptive BS-based optimization to improve navigation accuracy and efficiency.

3 Methodology

The methodology conducts data acquisition from RGB cameras and IMU combined with LiDAR sensors and GPS for UAV flight operations. Data preprocessing contains min-max normalization because it enhances model performance. Real-time obstacle detection with high precision and speed occurs through CYOLOv5 implementation. Flight paths are repeatedly modified through BS optimization to optimize the navigation process while reducing dangers in complicated airspace trajectories.

3.1 Data collection

UAV autonomous navigation dataset was gathered from a Kaggle source. The UAV autonomous navigation and obstacle avoidance sensor dataset is designed for investigation in deep learning-based autonomous flight and obstacle detection. The dataset includes 10,000 labeled UAV flight data samples recorded that integrate multi-sensor data including RGB cameras, LiDAR, IMU, and GPS to enable real-time path planning and obstacle avoidance. The gathered data was split with 80% training and 20 % testing. Figure 2 presents the data exploration.

Source: https://www.kaggle.com/datasets/ziya07/uav-autonomous-navigation-dataset/data

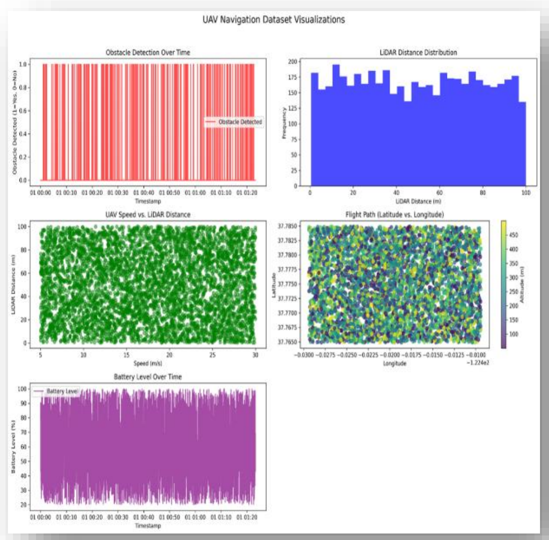

Figure 2: Data exploration

3.2 Data Pre-processing using min-max normalization

Min-Max normalization scales UAV navigation data, such as sensor inputs, to a range of [0, 1], enhancing deep learning model performance. It improves obstacle detection and path planning by ensuring consistent data distribution, reducing model bias, and accelerating convergence. The outputs or characteristics were linearly transformed from a particular range of values to another using this approach. The variables are often changed to fall among 0 and 1 or -1 and 1. Typically, the linear transformation provided as follows in Equation (1) is used to do the rescaling.

$$z = \frac{(w - \min(w))}{(\max(w) - \min(w))} \tag{1}$$

The max and min represent the lowest and greatest values in w, In other words, the range of w equals $\max(w) - \min(w)$. This normalization method's benefit stems from ensuring that every relationship in the data is precisely handled. This stage produced a uniformly scaled dataset with all sensor features (LiDAR, IMU, RGB, and GPS) normalized between 0 and 1, lowering feature dominance and enhancing training stability. Success was determined by comparing training convergence rates and finding lower initial loss values compared to unnormalized input, indicating more balanced feature contributions and smoother model learning.

3.2.1 Sensor fusion strategy

Multi-sensor integration process, the BS-CYOLOv5 framework fuses data from four sensors RGB camera, LiDAR, IMU, and GPS each contributing to different aspects of UAV perception and navigation as depicted in Figure 3. **RGB Camera**: Captures visual input, used as the primary data source for obstacle detection through YOLOv5. **LiDAR**: Provides depth perception and assists in spatial positioning of obstacles by estimating distances, especially useful in poor lighting conditions where RGB fails. **IMU**: Supplies inertial data (acceleration, rotation) to improve motion understanding and flight stabilization. **GPS**: Enables geospatial tracking, helping map detections to real-world coordinates and optimize flight trajectories.

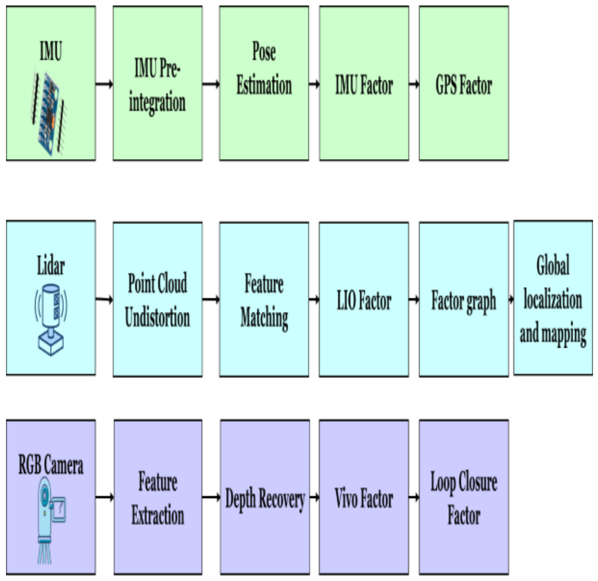

Figure 3: Sensor Fusion Strategy for BS-CYOLOv5 framework

3.3 BS-CYOLOv5

The BS-CYOLOv5 enhances UAV autonomous navigation and obstacle avoidance by integrating BS with CYOLOv5. BS is used to fine-tune YOLOv5 hyperparameters such as anchor box sizes, confidence threshold, and IoU threshold, which have an indirect impact on detection accuracy and reduce false positives. Furthermore, BS helps to optimize path planning by guiding adaptive UAV trajectory changes.

3.3.1 CYOLOv5 model

The YOLOv5 model improves autonomous UAV navigation by enabling the real-time obstacle detection using multi-sensor data. Its lightweight and fast architecture allows quick identification of obstacles, ensuring accurate and efficient navigation in dynamic environments. YOLOv5's high detection accuracy enhances UAV responsiveness, helping avoid collisions and navigate complex terrains effectively. A one-step technique that combines classification and bounding boxes into a regression issue is the YOLO object identification algorithm. It predicts the target's location in the grid by dividing photos through $T \times T$ meshes. These meshes are used to create bounding boxes with five parameters. To get the final predictions, non-maximum suppression is used to filter the prediction boxes. With two iterations, the YOLO line has grown quickly. The benchmark network for monitoring scenarios is the t version. The network was improved to the accuracy of the detection algorithm.

The most refined detection network of the YOLO object identification method is called YOLO v5, which uses K-means clustering, anchor box theory, and arithmetic set innovation to increase detection speed and accuracy. The term arithmetic set innovation refers to anchor forms that automatically learn through genetic mutation. With two incarnations, the YOLO line has expanded rapidly. This refers to the quick evolution between two major versions (YOLOv3 and YOLOv5), which boosted feature development. The 't' version of the network serves as a benchmark for monitoring scenarios. The 't' version denotes a small model variant designed for edge deployment. Figure 4 presents the network structure of CYOLOv5.

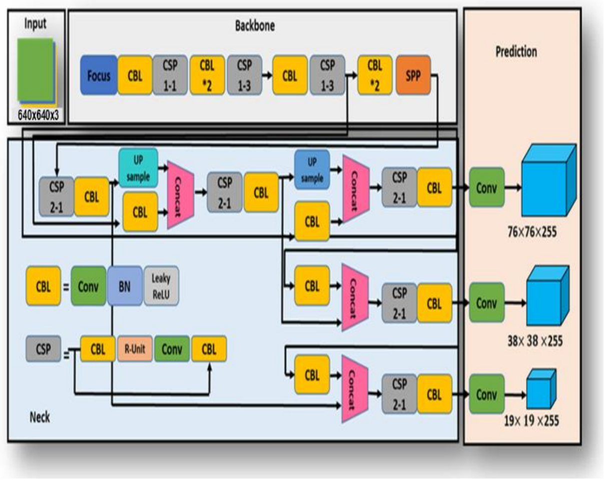

Figure 4: CYOLOv5 network structure

The YOLO v5 model makes use of an image input connection. It enhances the identification of tiny targets by using data. With a standardized input size of 640 x 640 x 3, the input image size is evenly scaled and supplied into the model for examination. YOLO v5's initial anchor frame is 116, 90, 156, 198, 373, 326, with a backbone network consisting of the focus structure. The focus structure slices the original image, creating a feature map of 320 × 320 × 12 and then applied 32 convolutional kernels. The transitions input features using two $1 * 1$ convolutions, improving CNN learning ability, breaking computing bottlenecks, and reducing memory costs.

The meshing concepts of the YOLO technique are carried over into YOLOv5. The size of the network input is 640 × 640 × 3. The detecting layer of the three scales large, medium, and small passes through the network with an output dimension of $T \times T \times m_b \times (s_w + s_z + s_x + s_g + s_p + m_d)$.

$$ \tag{2}$$

The split mesh's amount is $T \times T$. The number of pre-set previous boxes for every scale is shown by m_b. The number of classes that require prediction is denoted by m_d. Here, s_g refers to the number of geometric parameters per bounding box, which typically include center coordinates (x, y), width, and height and s_p is the number of prediction parameters per anchor box, including classification and regression outputs, such as bounding box offsets and object confidence scores. For instance, the detecting layer output dimensions of the network architecture of this dimension is 9600 in the large-scale T, m_b, and m_d. The parameters associated to the bounding box are s_w, s_z, s_x, and s_g. The bounding box reliability is s_{dj}, which stands for category I confidence. For these parameters to yield the final prediction box, Equation (3) must be decoded.

$$\begin{cases} a_x = \sigma(s_x) + d_w \\ a_g = \sigma(s_g) + d_z \\ a_w = o_w \cdot e^{s_w} \\ a_z = o_g \cdot e^{s_z} \\ score = \sigma(s_{dj}) \cdot max(\sigma(C_{cls})) \end{cases} \tag{3}$$

Where a_x and a_g are the predicted center coordinates of the bounding box. It is derived by applying the sigmoid activation to the raw network outputs s_x and s_g, then adding the grid cell offsets d_w and d_z to localize the prediction with the entire image. The width aw and height a_z are calculated by exponentiating the outputs s_w and s_z and scaling them by the anchor box dimensions o_w and o_g. The final detection score is calculated by combining the objectless confidence $s_{dj}$and the maximum class probability from C_{cls}, which are both passed through sigmoid activation.

For ground truth, the blue box is used. The label bounding box's width and height measurements are denoted by the letters $a_w, a_z, a_x, and\ a_g$, respectively. The distances d_w and d_z represent the distances between the top left corner of the grids and the grids that the label bounding box's center occupies. The anchor box is the red one. The width and height of the previous frame are represented by o_x and o_g, as shown in Equation (4).

$$K_{total} = \sum_{m=i}^{M} \sum_{i=1}^{A_j} \sum_{j=1}^{T_j} (K_{cls} + K_{CIOU}(a_w, a_z, a_x, a_g, o_g, o_x) + K_{offset}(d_w, d_z, w) + K_{conf}) \quad (4)$$

Where K represents the total training loss over all images, anchors, and targets. The algorithm includes classification loss (K_{cls}), localization loss (K_{CIOU}), objectness confidence loss (K_{conf}), and an offset penalty (K_{offset}) to improve spatial accuracy by minimizing displacement between the grid reference point and the actual object center. All variables employed in the loss terms, including coordinates, dimensions, offsets, and frame size, are consistent with typical object detection formulas.

➢ **Customized YOLOv5**

The performance of object detection algorithms is greatly impacted by positive-negative sample imbalance. Bounding box, class, and object loss are the three loss functions that are currently available. Generalized IoU loss is used as a bounding box loss function in YOLO v5. The Kclou loss number is based on the Complete IoU (CIoU) loss, which extends the conventional Intersection over Union (IoU) loss to account for center point distance and aspect ratio changes. Kclou is defined by Equations (5) to (7).

$$K_{CIoU} = 1 - I_{oU} + \frac{\rho^2(o, o^{gt})}{d^2} + \alpha u \quad (5)$$

$$I_{oU} = \frac{|B \cap A|}{|B \cup A|} \quad (6)$$

$$\alpha = \frac{u}{1 - J_{oU} + u} \quad (7)$$

IoU is the intersection-over-union score between predicted and ground-truth bounding boxes, $\rho^2(o, o^{gt})$ is the Euclidean distance between the box centers, d is the diagonal length of the smallest enclosing box, and $\alpha \cdot v$ is the aspect ratio penalty. This makes K_{CIoU} a more complete localization loss than IoU alone.

$$u = \frac{\left(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}\right)^2}{\pi^2} \quad (8)$$

The forecast box is B and the real box is A. The Euclidean distance is denoted by ρ. The prediction box's center point is denoted by o. o^{gt} is the target box's centre point. The minimal bounding rectangle's diagonal distance between the boxes is denoted by d. The weight function is denoted by α. The intersection ratio between the actual box and the predicted box is known as I_{oU}. The aspect ratios metric function is denoted by u. w^{gt} is the target box's width. The target box's height is denoted by h^{gt}. The recognition box's width is denoted by w and h is the prediction box's height. After this stage, the model effectively observed obstacles of various sizes and distances across UAV images using high-speed inference, resulting in bounding boxes and classification scores. Success was tested utilizing measures such as detection accuracy, precision, and IoU, and demonstrated strong performance under a variety of test situations.

3.3.2 BS

The BS enhances UAV autonomous navigation by optimizing path planning for real-time obstacle avoidance. BS uses a population-based search strategy, combining exploration and exploitation to adjust UAV trajectories dynamically. It minimizes path length and obstacle risk by adapting to environmental changes, ensuring efficient, accurate navigation with multi-sensor data integration. By using random crossover techniques and two selection procedures to choose the best population from existing and historical populations, BS is a population-based iteration that creates trial populations to regulate search direction amplitude. The five processes of BSA are initialization, selection I, mutation, crossover, and selection II, as follows.

➢ **Initialization**

Initial individuals O and initial old individuals $oldO$ are generated by utilizing BS, as shown in Equation (9).

$$O_{j,i} = low_i + rand(0,1).(up_i - low_i) \quad (9)$$

low_i and up_i denote the lower and upper boundaries of the ith dimension, ensuring that each variable remains with its feasible range. Here, $O_{j,i}$ and $oldO_{j,i}$, denote the $i - th$ dimensional values of the $j - th$ person in the current and previous populations. The function $rand(0,1)$ generates an equally distributed random number between $0\ and\ 1$, which is next linearly interpolated between the lower and upper bounds of each dimension. In this scenario, j refers to the index of an individual in the population size M, and i refers to the dimension index in the problem space size D. Each individual here represents a candidate hyperparameter set for YOLOv5, such as anchor box sizes, learning rate, batch size, and threshold values.

➢ **Selection I**

The choice of BS for every iteration starts with a process. Using population O and the historical population $oldO$, it seeks to reselect a new $oldO$ for determining the search direction. The "if-then" method is used to reselect the new $oldO$, as shown in Equation (10).

$$if\ b < a\ then\ oldO = O|b, a \sim V(0,1) \tag{10}$$

Where the update procedure is denoted by ≔, and random values from 0 to 1 are represented by a and b. The BS has storage because of the update process. $V(0,1)$ represents values within the range $[0,1]$ and The condition $b < a$ acts as a stochastic decision rule in the "if-then" construction. Following $oldO$ reselection, the individuals' rank is arbitrarily permuted by Equation (11) to reflect the present population (O). This helps to preserve diversity across hyperparameter combinations and prevents early convergence while tuning.

$$oldO \coloneqq permuting\ (OldO) \tag{11}$$

In Equation (11), the historical population $(OldO)$ is randomly permuted across people using the function permuting(·). This reorders the individuals (rows) in $oldO$. The "≔" reassignment allows the BS method to incorporate population variety and avoid premature convergence to local optimums during hyperparameter tuning.

➢ **Mutation**

The initial form of the experimental population is generated by the mutation operator N, which introduces unpredictability by changing individual search orientations. The amplitude control factor F affects the extent to which the current population is interrupted. Here, F is selected from a typical normal distribution, shown by $F \sim N(0,1)$, with a mean of 0 and a standard deviation of 1. This randomization ensures that the mutation step can traverse a large search space with varying intensities. The mutated vector is computed as shown in Equation (12).

$$W_j = O_j + F.(oldO_j - Q_j) \tag{12}$$

Where W_j is the mutated individual, O_j is the current individual, and Q_j is a randomly selected individual from the old population. The factor $F \sim N(0,1)$ dynamically controls the amplitude of the mutation. This technique balances exploration and convergence, increasing population variety during hyperparameter optimization in the BS-CYOLOv5 framework. This mutation procedure changes the hyperparameter values of YOLOv5, allowing it to explore various configurations for improved detection performance.

➢ **Crossover**

Both procedures create separate binary integer-valued vectors (map) of size $M \times C$ to choose which elements of individuals have to be changed.

In Strategy I, a mixing rule controls the number of elements updated in each individual. Each element is set to 1 (chosen for a crossover) if a randomly generated integer is smaller than the crossover rate, formally $mix[j] = 1$ if $rand(0,1)$ < rate, and 0 otherwise, for each dimension$j$.

In Strategy II (also referred to as Strategy J), exactly one element of each individual is randomly selected for crossover using$j = randi(0, C-1)$, where $randi(0, C-1)$ returns a random integer between 0 and C− 1, identifying the position to be changed.

Through the "if-then" rule, two techniques are equi-probably used to change the elements of individuals: if $map_{m,n} = 1$, where $m \in \{1,2, \ldots.., M\}$and $n \in \{1,2, \ldots.., C\}$, then, S is updated with $S_{m,n} \coloneqq O_{m,n}$. If certain users in O have exceeded the permitted search space restrictions after the crossover procedure, it must continue to be regenerated. This stage selectively changes specific YOLOv5 hyperparameters, incorporating advantageous qualities from multiple individuals to maximize detection-related setups.

➢ **Selection II**

The population O is updated using a greedy selection method in the BS selection II procedure, each trial individual (S_j) is compared to the matching individual (O_j) in the current population based on fitness values and t is the current iteration or generation number in the Backtracking Search (BS) algorithm. The fitness function $f(\cdot)$ assesses the objective performance of each solution using multi-objective criteria (e.g., detection accuracy, IoU score, and inference time). The population is updated with the use of Equation (13).

$$O_j^{(t+1)} = \begin{cases} S_j, if\ fitness\ (S_j) < fitness(O_j) \\ \qquad O_j^t, else \end{cases} \tag{13}$$

If the trial solution (S) has a lower fitness value than the current solution (O) for a minimization problem, it will be replaced by S_j in the following generation. Otherwise, the current solution O_j^t is retained. The fitness function here assesses each collection of hyperparameters based on detection accuracy, IoU, and inference speed, ensuring that only the best configurations are retained for YOLOv5 performance. This stage produced an optimal set of YOLOv5 hyperparameters (confidence threshold, anchor sizes, IoU threshold, learning rate), as well as refined UAV flight paths that reduced risk and distance. After adding BS, the model demonstrated increased generalization, faster convergence, and fewer false positives. Table 2 depicts the Hyperparameter Tuning Summary for BS-CYOLOv5. Algorithm 1 depicts the pseudocode for the BS-CYOLOv5 framework and Figure 5 depicts the diagram for BS-CYOLOv5 framework.

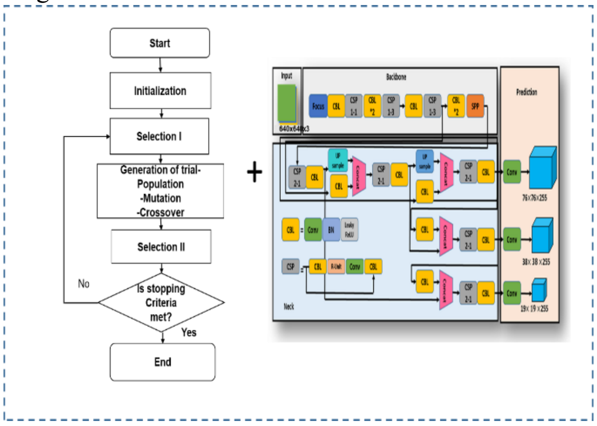

Figure 5: Diagram for BS-CYOLOv5 framework.

Table 2: Hyperparameter tuning summary for BS-CYOLOv5

Hyperparameter	**Initial Value**	**Tuned Value (by BS)**	**Role in Model**	**Impact on Performance**
Anchor Box Sizes	[116, 90, 156,198, 373,326]	[72,54, 142,128, 310,290]	Defines bounding box shapes for different object scales	Improves detection of varying obstacle sizes
Learning Rate	0.01	0.005	Controls optimizer step size	Stabilizes training, improves convergence and accuracy
Batch Size	16	32	Number of samples processed per update	Enhances generalization and training efficiency
Confidence Threshold	0.25	0.18	Minimum confidence to keep prediction	Reduces false positives, increases precision
IoU Threshold	0.5	0.65	Used in NMS to filter overlapping boxes	Boosts recall, ensures spatial overlap accuracy
Composite Fitness Score	-	Accuracy: 98.1%, IoU: 96.1%, Inference Time: 17.4 ms	Guides BS tuning by balancing multiple evaluation criteria	Achieves optimal detection quality and speed simultaneously

Algorithm 1: Backtracking Search-mutated Customized YOLOv5 (BS-CYOLOv5)

```
Input: Objective function $f(x)$, population size $M$, dimension $C$, maximum iterations $T$
Output: Optimal solution $x\_best$
1: Initialize population $O$ using Equation (9)
2: Set historical population $Q = O$
3: for $t = 1$ to $T$ do
4:     Compute mutation amplitude $E$ using Equation (11)
5:     Generate trial vector $S$ using crossover mask (Equation 12)
6:     Evaluate the fitness of original and trial individuals
7:     Apply selection (Equation 13) to update $O$
8: end for
9: Return best solution $x\_best$ from $O$
```

4 Result

The experiment was conducted using Python 3.12 which is operated on Windows 11 with a 7th generation Core i7 processor and 32 GB of RAM and an NVIDIA GeForce RTX 3060 GPU with 12 GB VRAM. The modern laptop design facilitated multitasking and performance evaluation during development, making experimental tasks more efficient and manageable. The result comparison parameters, such as accuracy, f1-score, recall precision and IOU are used to demonstrate the comparison of the suggested framework BS-CYOLOv5 with the traditional models [21].

4.1 Confusion matrix

This confusion matrix shows perfect performance in UAV obstacle detection. The UAV correctly identified 4764 no-obstacle instances and 236 obstacle instances, with zero false positives or false negatives. This result indicates an ideal detection system, enhancing UAV navigation efficiency and safety for real-time obstacle avoidance and autonomous path planning. Figure 6 depicts the confusion matrix for UAV obstacle detection under ideal test conditions.

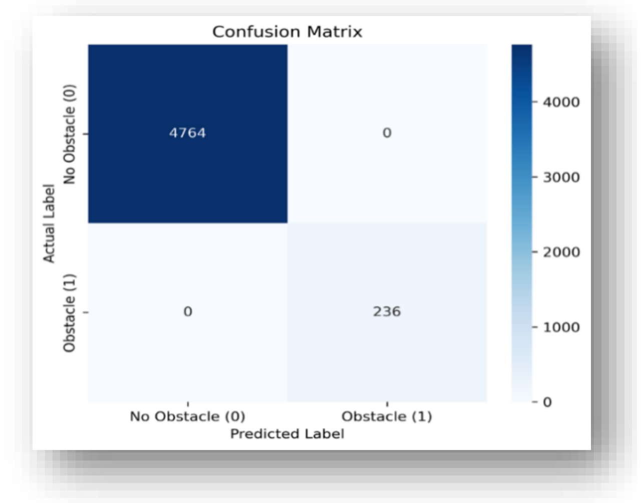

Figure 6: Confusion matrix for UAV obstacle detection under ideal test conditions

The UAV successfully detected 4764 no-obstacle situations and 236 obstacle situations. The UAV identified the obstacles without producing any false negatives or false positives. This suggests that, this assessment has 100 % accuracy, precision, and recall. The subset in Figure 4 displays the best case performing under ideal scenarios, whereas the total test accuracy is a more realistic representation of the model when applied in the context of variable UAV flight scenarios.

4.2 Altitude Variation over Time

It is defined as the change in vertical position over time when flying. In the research, this variable is utilized to assess the stability and accuracy of the UAV's altitude control system while navigating autonomously. The observed altitude variation could note the UAV's normal flight adjustments due to environmental conditions or navigation adjustments. Minor altitude variations are expected with autonomous UAV navigation and are not indicative of relevant instability. In this experiment, it is noted that the UAV was able to control altitude satisfactorily without extreme or erratic deviations. However, ongoing improvement of multi-sensor fusion and adaptive control systems potentially improves altitude accuracy in more variable or turbulent environments. Figure 7 shows the altitude variation over time during UAV flight and presents slight variations around a functional operating range.

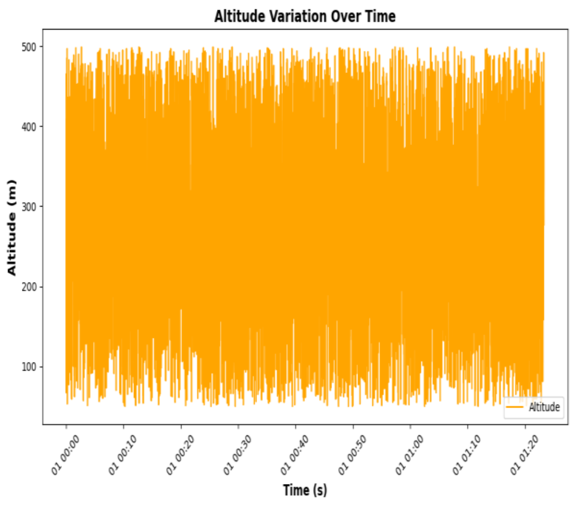

Figure 7: UAV altitude variation over time during navigation.

Figure 7 depicts the UAV's altitude variation over time, with fast fluctuations ranging from roughly 50 meters to 500 meters over the observation period. The time axis spans around 1 hour and 20 minutes, with data recorded at high frequency. The graphic depicts the system's dynamic responsiveness to flight conditions while remaining within a controlled altitude range. This displays how well the BS-CYOLOv5 model performs in maintaining stable vertical navigation.

4.3 Accuracy

The accuracy of autonomous navigation and obstacle avoidance systems for UAVs involves correctly identifying obstacles and planning optimal paths to minimize navigation errors. Table 3 depicts the accuracy comparison between models for UAV obstacle detection performance. Figure 8 depicts the accuracy that comparing SSD MobileNet and BS-CYOLOv5

Table 3: Accuracy comparison between models for UAV obstacle detection performance

Methods	**Accuracy (%)**
SSD MobileNet [18]	96.75%
BS-CYOLOv5 [Proposed]	**98.10%**

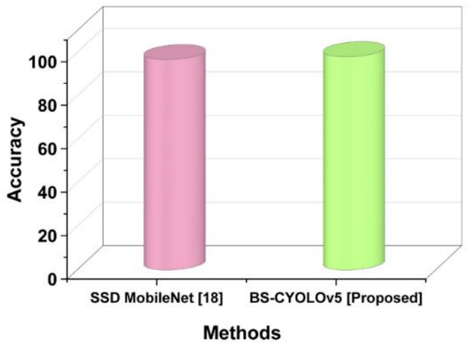

Figure 8: Accuracy bar chart comparing SSD MobileNet and BS-CYOLOv5

Accuracy refers to the overall correctness of the model's predictions. The suggested BS-CYOLOv5 obtains the best accuracy (98.10%), outperforming SSD MobileNet (96.75%). This demonstrates higher overall classification and detection reliability.

4.4 Precision

Precision refers to the proportion of successfully identified barriers among all detected by the model. It shows how successfully BS-CYOLOv5 minimizes false positives when detecting barriers. High precision means that the majority of detected barriers are accurate. This is crucial in UAV navigation to avoid unwanted movements. Table 4 represents the Precision values across models showing obstacle classification effectiveness. Figure 9 depicts the comparison of model precision highlighting proposed BS-CYOLOv5's performance

Table 4: Precision values across models showing obstacle classification effectiveness

Methods	**Precision (%)**
SSD MobileNet [18]	95.89
UAD-YOLOv8 [19]	80.7
BS-CYOLOv5 [Proposed]	**97.52**

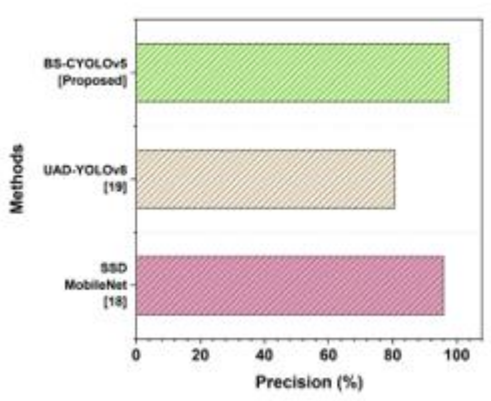

Figure 9: Comparison of model precision highlighting proposed BS-CYOLOv5's performance

Precision refers to the proportion of real obstacle detections among all positive forecasts. BS-CYOLOv5 achieves 97.52% precision, outperforming SSD MobileNet (95.89%) and UAD-YOLOv8 (80.7%), demonstrating that false positives are successfully reduced.

4.5 Recall

Recall measures the model's ability to detect all genuine obstacles in the environment. It demonstrates how effectively BS-CYOLOv5 avoids missing any threats and reduces false negatives. High recall guarantees safe UAV operation by detecting all important impediments. Table 5 represents the Recall comparison across models for detecting true positive obstacles. Figure 10 illustrates the Recall performance chart comparing BS-CYOLOv5 and existing models

Table 5: Recall comparison across models for detecting true positive obstacles

Methods	**Recall (%)**
SSD MobileNet [18]	97.22
UAD-YOLOv8 [19]	73.8
BS-CYOLOv5 [Proposed]	**97.54**

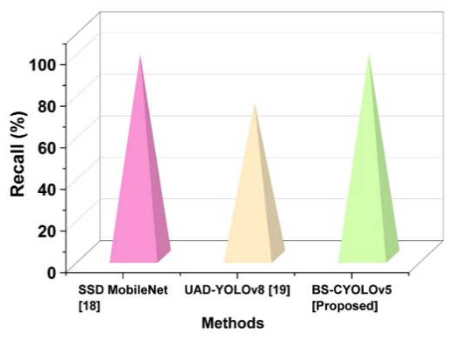

Figure 10: Recall performance chart comparing BS-CYOLOv5 and existing models

Recall measures the capacity to detect all genuine barriers. The suggested technique achieves a recall of 97.54%, slightly higher than SSD MobileNet (97.22%) but much better than UAD-YOLOv8 (73.8%), indicating fewer missing obstacles.

4.6 F1-Score

The F1-score is the harmonic mean of precision and recall, which balances false positives and false negatives. BS-CYOLOv5 introduces a single, robust metric for assessing detection performance. A high F1-score shows that the model retains its accuracy and completeness. It is useful when precision and recall are equally essential. Table 6 depicts the F1-score results showing the balance between precision and recall accuracy. Figure 11 represents the F1-score visualization comparing the proposed model to baselines.

Table 6: F1-score results showing the balance between precision and recall accuracy

Methods	**F1-score (%)**
SSD MobileNet [18]	96.41
UAD-YOLOv8 [19]	0.771
BS-CYOLOv5 [Proposed]	**97.28**

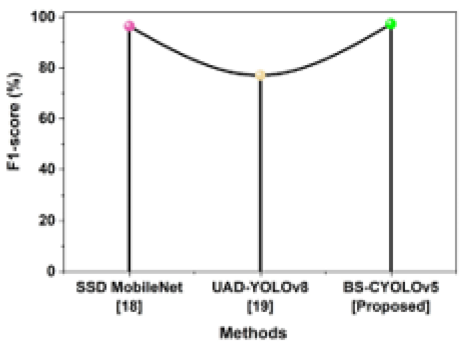

Figure 11: F1-score visualization comparing the proposed model to baselines

The F1-score integrates precision and recall into a single performance metric. BS-CYOLOv5 scores 97.28%, overcoming SSD MobileNet (96.41%) and UAD-YOLOv8 (0.771), indicating good detection consistency.

4.7 Intersection over Union (IoU)

IoU calculates the overlap between expected and ground truth bounding boxes. It is used in BS-CYOLOv5 to assess localization accuracy during the training and prediction phases. A higher IoU suggests improved spatial alignment of detected obstacles. Table 7 depicts the IoU comparison indicating spatial accuracy of predicted bounding boxes. Figure 12 shows the IoU results visualization showing localization precision across models.

Table 7: IoU comparison indicating spatial accuracy of predicted bounding boxes

Methods	**IOU (%)**
CNN [20]	89.9
BS-CYOLOv5 [Proposed]	**96.1%**

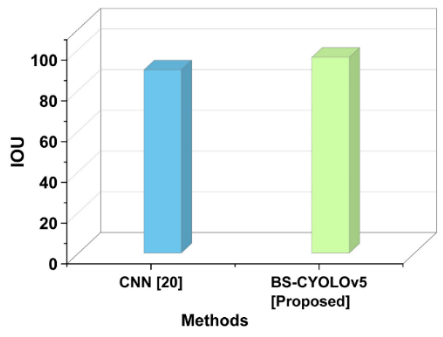

Figure 12: IoU results visualization showing localization precision across models

The BS-CYOLOv5 model has an IoU of 96.1%, which is higher than CNN's IOU value of 89.9%, showing superior spatial accuracy in obstacle location.

4.8 5-Fold cross-validation

5-Fold Cross-Validation is a strategy for evaluating a model's generalizability and robustness. It involves partitioning the dataset into five equal parts, training the model on four of them, and then testing it on the remaining one five times. This ensures that each data point is used for both training and validation, lowering the danger of overfitting and producing a more accurate performance estimate. It confirms the consistency and reliability of the BS-CYOLOv5 model across various data subsets when detecting UAV obstacles. Table 7 represents an evaluation of BS-CYOLOv5 using five-fold validation metrics.

Table 7: Evaluation of BS-CYOLOv5 Using Five-Fold Validation Metrics

Folds	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**	**IoU (%)**
1	97.85	97.20	97.35	97.27	95.80
2	98.00	97.45	97.48	97.46	96.05
3	97.92	97.35	97.40	97.37	96.00
4	97.95	97.50	97.55	97.52	96.10
5	**98.10**	**97.52**	**97.54**	**97.28**	**96.10**

The BS-CYOLOv5 model performed effectively and consistently over all five folds. The accuracy ranged from 97.85% to 98.10%, with an average of 97.96%. Precision and recall remained consistent at 97.40% and 97.46%, respectively, while the F1-score averaged 97.38% and IoU remained high at 96.01%. These results support the model's robust and precise obstacle detection in UAV autonomous navigation.
The data is split into multiple trial evaluations such as 80% training and 20% Testing and 70% training and 30% Testing. Table 8 shows the outcomes of the multiple trial evaluations for the proposed BS-CYOLOv5 model.

Table 8: Outcomes of the multiple trial evaluations for the proposed BS-CYOLOv5 model

Methods	**Training and Testing**	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-score (%)**	**IOU (%)**
BS-CYOLOv5 [Proposed]	80% training and 20% Testing	98.10	97.52	97.54	97.28	96.1
	70% training and 30% Testing	97.50	97.14	96.81	96.92	95.87

The BS-CYOLOv5 model was tested using two distinct train-test splits to determine its robustness and generalizability. When trained with 80% of the data and tested with 20%, it achieved 98.10% accuracy, 97.52% precision, 97.54% recall, 97.28% F1-score, and 96.1% IoU, indicating high reliability in obstacle detection. With a 70% reduction in training and a 30% increase in testing, performance declined marginally, providing 97.50% accuracy, 97.14% precision, 96.81% recall, 96.92% F1-score, and 95.87% IoU. These results demonstrate that the model performs effectively even when there is a smaller amount of training data.

In UAV autonomous navigation and obstacle avoidance, BS-CYOLOv5 outperforms Q-Learning in complex environments. For static and 2 dynamic obstacles, BS-CYOLOv5 achieved a training time (150.782s vs. 336.5344s), the shortest distance (57 vs. 60), and the longest distance (125 vs. 131). For static and 4 dynamic obstacles, BS-CYOLOv5 also excelled with a training time (180.921s vs. 357.3923s), the shortest distance (66 vs. 70). BS-CYOLOv5 also performed better in terms of the longest distance travelled, recording 110 units compared to 112 units by the Q-Learning algorithm, indicating a more efficient navigation path. Table 9 presents the time, the shortest and longest distance of the proposed model.

Table 9: Time, shortest and longest distance of proposed model

Method	**Training Time(s)**	**Shortest distance**	**Longest distance**
Q-Learning Algorithm(Static + 2Dynamic Obstacles) [21]	336.5344	60	131
BS-CYOLOv5 (Static + 2 Dynamic Obstacles) [proposed]	150.782	57	125
Q-Learning Algorithm (Static + 4 Dynamic Obstacles) [21]	357.3923	70	112
BS-CYOLOv5 (Static + 4 Dynamic Obstacles) [proposed]	180.921	66	110

5 Discussion

Autonomous navigation and obstacle avoidance enable UAVs to navigate complex environments independently, using AI and sensor data to detect obstacles, optimize paths, and ensure safe, efficient flight. Existing UAV navigation and obstacle avoidance methods have severe limitations. NMPC and Faster R-CNN [9][10] have computational limitations and lack depth understanding. SAC-based models [11][12] are expensive and imprecise, whereas Q-Learning and SARSA [13] exhibit slow convergence and energy inefficiency. DRL-based approaches [14] are resource-intensive, while A-based planners [15] have memory restrictions. DAC-SAC [16] suffered with insufficient training data. SSD MobileNet [18] and UAD-YOLOv8 [19] have reduced precision and recall, while CNN models [20] are not adaptable in dynamic, real-world UAV environments, affecting obstacle detection performance. Q-Learning [21], being model-free, requires extensive exploration to converge, leading to slow learning in high-dimensional state spaces. It lacked adaptability to rapidly changing environments and can suffer from the curse of dimensionality, making real-time decision-making for UAV navigation inefficient in complex and dynamic scenarios. The proposed BS-CYOLOv5 outperforms the method used in [12], which uses Q-Learning and SARSA and achieves an estimated accuracy of ~95%, precision of ~94.1%, recall of ~94.3%, F1-score of ~94.2%, and IoU of ~92.5%, across all measures. Specifically, BS-CYOLOv5 achieves 98.10% accuracy, 97.52% precision, 97.54% recall, 97.28% F1-score, and 96.1% IoU, indicating significant gains in obstacle detection accuracy, false positive reduction, and improved localization. Real-world applications require UAVs to navigate unpredictable settings with fluctuating lighting, dynamic impediments, and sensor noise. The proposed BS-CYOLOv5 provides reliable obstacle detection and adaptive path planning for deployment beyond controlled simulations. Although the experiments were carried out in a simulated setting, the UAV situations were created using the AirSim simulator, which included real-world physics and multi-sensor input. However, ambient noise and sensor variability could cause a domain gap when switching to outdoor environments. Future work will incorporate real-world validation to address this gap.

6 Conclusion

The effectiveness of a DL-based framework for enhancing the autonomous navigation and obstacle avoidance capabilities of UAVs in dynamic environments. By integrating multi-sensor data from RGB cameras, LiDAR, IMU, and GPS, the proposed system enables real-time obstacle detection using the CYOLOv5 model, which achieves high detection accuracy and speed across diverse environmental conditions. The BSA further enhances navigation by dynamically adjusting flight paths, ensuring efficient and collision-free operation. Experimental results demonstrated notable improvements in obstacle detection of 98.10% accuracy, 97.52% precision, 97.54% recall, 97.28% F1-score, and 96.1% IoU. These findings highlight how combining advanced DL techniques with optimization algorithms strengthens UAV adaptability and operational safety in complex scenarios. Limitations include challenges in handling extreme weather conditions, sensor noise, and computational constraints during real-time processing. While the results appear promising, more testing is needed to determine the model's generalizability to new situations or UAV platforms. Future research will include real time validation to assess the resilience of each framework component and quantify performance across a variety of operating scenarios. Future advancement will focus on enhancing the framework's scalability across different UAV models, improving sensor fusion techniques, and refining adaptability to unpredictable environmental changes.

References

[1] Mohsan S A H, Othman N Q H, Li Y, Alsharif M H, Khan M A (2023). Unmanned aerial vehicles (UAVs): Practical aspects, applications, open challenges, security issues, and future trends. *Intelligent Service Robotics*, 16(1), 109–137. https://doi.org/10.1007/s11370-022-00452-4

[2] Thangamani R, Suguna R K, Kamalam G K (2024). Drones and autonomous robotics incorporating computational intelligence. In *Computational Intelligent Techniques in Mechatronics*, 243–296. https://doi.org/10.1002/9781394175437.ch8

[3] Fei W A N G, Xiaoping Z H U, Zhou Z, Yang T A N G (2024). Deep-reinforcement-learning-based UAV autonomous navigation and collision avoidance in unknown environments. *Chinese*

Journal of Aeronautics, 37(3), 237–257. https://doi.org/10.1016/j.cja.2023.09.033

[4] Lu X, Yang Z, Yang Y, Sharma A (2021). Research on estimation of paddy field area index based on UAV remote sensing images. *Informatica*, 45(5). https://doi.org/10.31449/inf.v45i5.3558

[5] Khan A, Gupta S, Gupta S K (2022). Emerging UAV technology for disaster detection, mitigation, response, and preparedness. *Journal of Field Robotics*, 39(6), 905–955. https://doi.org/10.1002/rob.22075

[6] Martinez V C, Ince B, Selvam P K, Petrunin I, Seo M, Anastassacos E, Knorr S (2021). Detect and avoid considerations for safe sUAS operations in urban environments. In *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, 1–10. https://doi.org/10.1109/DASC52595.2021.9594407

[7] Dai M, Huang N, Wu Y, Gao J, Su Z (2022). Unmanned-aerial-vehicle-assisted wireless networks: Advancements, challenges, and solutions. *IEEE Internet of Things Journal*, 10(5), 4117–4147. https://doi.org/10.1109/JIOT.2022.3230786

[8] Kaleem Z, Orakzai F A, Ishaq W, Latif K, Zhao J, Jamalipour A (2024). Emerging trends in UAVs: From placement, semantic communications to generative AI for mission-critical networks. *IEEE Transactions on Consumer Electronics*. https://doi.org/10.1109/TCE.2024.3434971

[9] Lindqvist B, Mansouri S S, Haluška J, Nikolakopoulos G (2021). Reactive navigation of an unmanned aerial vehicle with perception-based obstacle avoidance constraints. *IEEE Transactions on Control Systems Technology*, 30(5), 1847–1862. https://doi.org/10.1109/TCST.2021.3124820

[10] Lee H Y, Ho H W, Zhou Y (2021). Deep learning-based monocular obstacle avoidance for unmanned aerial vehicle navigation in tree plantations: Faster region-based convolutional neural network approach. *Journal of Intelligent & Robotic Systems*, 101(1), 5. https://doi.org/10.1007/s10846-020-01284-z

[11] Pedro D, Matos-Carvalho J P, Fonseca J M, Mora A (2021). Collision avoidance on unmanned aerial vehicles using neural network pipelines and flow clustering techniques. *Remote Sensing*, 13(13), 2643. https://doi.org/10.3390/rs13132643

[12] Xue Z, Gonsalves T (2021). Vision based drone obstacle avoidance by deep reinforcement learning. *AI*, 2(3), 366–380. https://doi.org/10.3390/ai2030023

[13] Tu G T, Juang J G (2023). UAV path planning and obstacle avoidance based on reinforcement learning in 3D environments. *Actuators*, 12(2), 57. https://doi.org/10.3390/act12020057

[14] Ouahouah S, Bagaa M, Prados-Garzon J, Taleb T (2021). Deep-reinforcement-learning-based collision avoidance in UAV environment. *IEEE Internet of Things Journal*, 9(6), 4015–4030. https://doi.org/10.1109/JIOT.2021.3118949

[15] Tullu A, Endale B, Wondosen A, Hwang H Y (2021). Machine learning approach to real-time 3D path planning for autonomous navigation of unmanned aerial vehicle. *Applied Sciences*, 11(10), 4706. https://doi.org/10.3390/app11104706

[16] Gao Y, Ren L, Shi T, Xu T, Ding J (2024). Autonomous obstacle avoidance algorithm for unmanned aerial vehicles based on deep reinforcement learning. *Engineering Letters*, 32(3).

[17] Almseidein T A, Alzidaneen A (2025). Optimizing UAV trajectories with multi-layer artificial neural networks. *Informatica*, 49(2). https://doi.org/10.31449/inf.v49i2.7178

[18] RS R, Al-Shehari T, Nathan S, Alfakih T, Alsalman H (2024). An unmanned aerial vehicle captured dataset for railroad segmentation and obstacle detection. *Scientific Data*, 11(1), 1–12. https://doi.org/10.1038/s41597-024-03952-3

[19] Du Z, Feng X, Li F, Xian Q, Jia Z (2024). A lightweight UAV visual obstacle avoidance algorithm based on improved YOLOv8. *Computers, Materials & Continua*, 81(2). http://dx.doi.org/10.32604/cmc.2024.056616

[20] Wang D, Li W, Liu X, Li N, Zhang C (2020). UAV environmental perception and autonomous obstacle avoidance: A deep learning and depth camera combined solution. *Computers and Electronics in Agriculture*, 175, 105523. https://doi.org/10.1016/j.compag.2020.105523

[21] Sonny A, Yeduri S R, Cenkeramaddi L R (2023). Q-learning-based unmanned aerial vehicle path planning with dynamic obstacle avoidance. *Applied Soft Computing*, 147, 110773. https://doi.org/10.1016/j.asoc.2023.110773