machines

Article

An Integrated YOLOv5 and Hierarchical Human-Weight-First Path Planning Approach for Efficient UAV Searching Systems

Ing-Chau Chang [1,*], Chin-En Yen [2], Hao-Fu Chang [1], Yi-Wei Chen [1], Ming-Tsung Hsu [1], Wen-Fu Wang [1], Da-Yi Yang [1] and Yu-Hsuan Hsieh [1]

1 Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua 50007, Taiwan; fchang@wisdome.ai (H.-F.C.); yiwei.chen@g.ncu.edu.tw (Y.-W.C.); xmc510063@gapp.nthu.edu.tw (M.-T.H.); wang1020351@gapp.nthu.edu.tw (W.-F.W.); m1254004@gm.ncue.edu.tw (D.-Y.Y.); s0954010@mail.ncue.edu.tw (Y.-H.H.)

2 Department of Early Childhood Development and Education, Chaoyang University of Technology, Taichung 41349, Taiwan; ceyen@cyut.edu.tw

* Correspondence: icchang@cc.ncue.edu.tw; Tel.: +886-4-723-2105

Abstract: Because the average number of missing people in our country is more than 20,000 per year, determining how to efficiently locate missing people is important. The traditional method of finding missing people involves deploying fixed cameras in some hotspots to capture images and using humans to identify targets from these images. However, in this approach, high costs are incurred in deploying sufficient cameras in order to avoid blind spots, and a great deal of time and human effort is wasted in identifying possible targets. Further, most AI-based search systems focus on how to improve the human body recognition model, without considering how to speed up the search in order to shorten the search time and improve search efficiency, which is the aim of this study. Hence, by exploiting the high-mobility characteristics of unmanned aerial vehicles (UAVs), this study proposes an integrated YOLOv5 and hierarchical human-weight-first (HWF) path planning framework to serve as an efficient UAV searching system, which works by dividing the whole searching process into two levels. At level one, a searching UAV is dispatched to a higher altitude to capture images, covering the whole search area. Then, the well-known artificial intelligence model YOLOv5 is used to identify all persons in the captured images and compute corresponding weighted scores for each block in the search area, according to the values of the identified human bodies, clothing types, and clothing colors. At level two, the UAV lowers its altitude to sequentially capture images for each block, in descending order according to its weighted score at level one, and it uses the YOLOv5 recognition model repeatedly until the search target is found. Two improved search algorithms, HWFR-S and HWFR-D, which incorporate the concept of the convenient visit threshold and weight difference, respectively, are further proposed to resolve the issue of the lengthy and redundant flight paths of HWF. The simulation results suggest that the HWF, HWFR-S, and HWFR-D search algorithms proposed in this study not only effectively reduce the length of a UAV's search path and the number of search blocks but also decrease the search time required for a UAV to locate the search target, with a much higher search accuracy than the two traditional search algorithms. Moreover, this integrated YOLOv5 and HWF framework is implemented and tested in a real scenario to demonstrate its capability in enhancing the efficiency of a search and rescue operation.

Keywords: unmanned aerial vehicle; hierarchical human-weight-first path planning; artificial intelligence image recognition; YOLOv5; searching corners without cameras

Citation: Chang, I.-C.; Yen, C.-E.; Chang, H.-F.; Chen, Y.-W.; Hsu, M.-T.; Wang, W.-F.; Yang, D.-Y.; Hsieh, Y.-H. An Integrated YOLOv5 and Hierarchical Human-Weight-First Path Planning Approach for Efficient UAV Searching Systems. *Machines* **2024**, *12*, 65. https://doi.org/10.3390/machines12010065

Academic Editors: Maria Eusebia Guerrero-Sanchez and Omar Hernández-González

Received: 3 December 2023
Revised: 6 January 2024
Accepted: 8 January 2024
Published: 16 January 2024

1. Introduction

As the technology of the unmanned aerial vehicle (UAV) has seen significant progress in recent years, a number of applications have been proposed for it [1,2] due to its unique characteristics, such as higher mobility and more flexible integration with different equipment, such as sensors and cameras, etc. [3,4]. The researchers of [5] explore algorithms for the formation movement of UAV swarms, with the objective of facilitating simultaneous adjustments to the formation shape while the UAV swarm is in motion. Signal transmission is another highly significant topic in UAV control. The research in [6] proposes automatic modulation classification utilizing deep learning in this context. The study in [7] addresses improvements to existing GNSS systems, such as GPS positioning, tackling issues related to inaccuracies. The researchers propose a time-differenced carrier phase (TDCP) derivation-controlled GNSS/IMU integration scheme to successfully acquire vehicle information such as the relative position and heading. Real-world tests demonstrate that this method exhibits higher accuracy compared to traditional algorithms. In recent years, the increasing integration of UAVs with various interdisciplinary domains has also been observed. Koopman operators are mathematical tools used to describe the evolution of nonlinear dynamic systems. The work of [8] proposes robust tube-based model predictive control with Koopman operators, while [9] integrates Koopman operators with the control of UAVs. Furthermore, there exist various UAV path planning problems and related studies, such as the capacitated arc routing problem (CARP). The objective of CARP is to find the shortest path in a mixed graph with undirected edges and directed arcs, minimizing the distance of the path while considering capacity constraints for objects moving on the graph. In [10], the study introduces a memetic algorithm based on Two_Arch2 (MATA), which simultaneously considers multiple optimization objectives for the path planning problem, including the total cost, makespan, carbon emissions, and load utilization rate.

Recently, UAVs have been used for search and rescue (SAR) missions to find missing persons at the scene of a natural disaster or when an emergency event occurs [11–13]. The issue of missing persons is a challenging societal problem, particularly when involving minors. Children, due to their smaller stature, are susceptible to disappearance within large crowds, especially in crowded places such as amusement parks, making it difficult to notice their absence. Unfortunately, they generally exhibit a lower level of vigilance towards unfamiliar individuals, rendering them vulnerable to abduction. As the duration of a missing person's search is prolonged, the probability of encountering a perilous situation escalates, imposing significant psychological distress upon parents.

However, there is a limited amount of research aimed at identifying specific individuals, such as missing persons, and researchers have primarily relied on fixed cameras installed in specific areas. This limitation prevents the continuous tracking of targets, leading to difficulties in inferring their actual positions due to the limited perspective and potential blind spots. Furthermore, most of the existing works on search and rescue adopt unmanned aerial vehicles (UAVs) [14–16] and employ indiscriminate search algorithms, without prioritizing the areas where the search target may be located, resulting in inefficient search operations and excessive UAV power consumption. Hence, intelligent approaches such as AI-enabled object detection in UAVs [17] are proposed to overcome this problem.

One of the major research directions regarding SAR involves coordinating multiple UAVs to cover multiple regions, which may be scattered or not [18–21]. These approaches focus on how to balance the time consumption of UAVs or design effective coverage flights to cover the search area completely, with no collisions of multiple UAVs. On the other hand, this study aims to improve the search time and costs related to an assigned search area for a single UAV. It addresses the inefficiency problem of SAR mentioned above by proposing an integrated YOLOv5 and hierarchical HWF path planning approach. This integrated approach executes the following steps. First, the user specifies the geographical coordinates of the search area and important features of the search target. Then, the searching UAV is dispatched to a higher altitude to capture images covering the whole search area using an onboard camera. Next, the search area is partitioned into multiple blocks. The Jetson Nano mounted on the searching UAV captures and transmits images to a cloud server through wireless communication. On the cloud server, the well-known artificial intelligence model YOLOv5 is used for human body recognition and clothing recognition, and the KNN algorithm is used to identify clothing colors. Corresponding weighted scores are computed with the identified human bodies, clothing types, and clothing colors within each block. After this, the cloud server issues a command to the UAV to adjust its coordinates and lower its altitude to capture clearer images that cover a single block. The proposed human-weight-first (HWF) path planning algorithm is utilized to guide the UAV to visit blocks sequentially, according to their weighted scores, in descending order. YOLOv5 and the KNN algorithm are iteratively employed on the cloud server side to recognize human bodies, clothing types, and clothing colors for images captured at lower altitudes. Upon confirming the presence of the search target, the human body image and its location are reported to the user, which concludes the UAV's search mission. This integrated YOLOv5 and hierarchical HWF path planning approach can prioritize the block where the search target is most likely to be located, to shorten the search time and costs, which significantly improves the search efficiency and avoids the searching of corners without cameras.

Consequently, this study achieves the following contributions.

1. It utilizes the existing YOLOv5 model to automatically recognize the search target and uses KNN color recognition to recognize clothing/pant colors in real time; it thus avoids wasting time and human effort in the manual identification of possible targets.
2. According to the recognition results of the YOLOv5 model and KNN color recognition, the study proposes a weighting subroutine to calculate the human weights of each block and the hierarchical human-weight-first (HWF) path planning algorithm to dispatch the UAV to capture images repeatedly of the search area and each block at different altitudes.
3. It proposes a complete flowchart of the integrated YOLOv5 and HWF framework to reduce the search time and avoid searching corners without cameras.

2. Related Work

2.1. Traditional Unmanned Aerial Vehicle Path Planning Methods for Search and Rescue Operations

Several search and rescue methods have been proposed recently [14–16]. In [14], the sweep line search method conducts a thorough search from left to right, as illustrated in Figure 1. Meanwhile, ref. [15] introduces the spiral search, which navigates the designated search area in a spiral pattern, as depicted in Figure 2. Both methods are uncomplicated and exhibit algorithms with linear time complexity in relation to the search area. Differing from these two methods, refs. [16,22] introduce block-based methods. These approaches offer the advantage of categorizing the whole search area into blocks with and without search targets. Figure 3 demonstrates the relationship between the UAV's perspective and the altitude concerning the search blocks when the whole search area is partitioned [22]. Through the traveling salesman problem (TSP) [23] approach, the shortest path that does not require the visiting of all blocks is computed if all blocks with search targets have been recognized in advance. However, the four methods mentioned above do not prioritize the block searching sequence in proximity to the search target, which results in inadequate search efficiency. Therefore, taking inspiration from block-based approaches, this study assigns priority to all blocks based on the likelihood of the blocks containing potential targets, which are automatically recognized in real time using the YOLOv5 model. In contrast to [16], which primarily focuses on finding the shortest path, this study emphasizes improving the search efficiency to yield the shortest search time by searching in the block with the highest priority first.

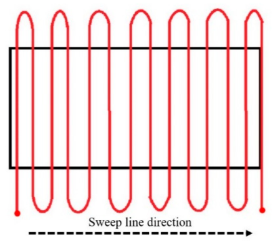


Figure 1. Sweep line search.

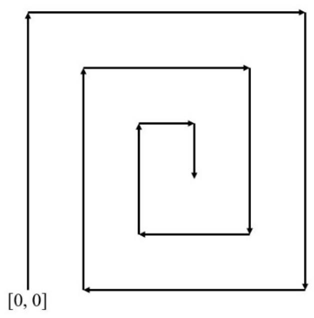


Figure 2. Spiral search.

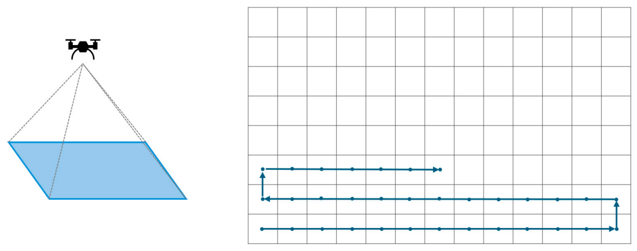


Figure 3. The relationship between the altitude of the UAV and the partitioned search area.

2.2. Search Target Recognition Techniques

2.2.1. Color Space Exchange

The RGB color space is the most widely used color space, where RGB denotes red, green, and blue. It is similar to the well-known concept of the primary colors of light, where mixing these colors yields various levels of brightness and chromaticity. However, the RGB color space has a strong dependence on the lighting conditions, meaning that the color of an object can change with variations in brightness. In addition, the three elements in the RGB color space are highly correlated, indicating that a change in one element will result in a corresponding change in the perceived color. Therefore, using the RGB color space for the color extraction of objects is not ideal [24]. In contrast, the HSV color space [25] is more intuitive and easily understood compared to the RGB color space. It separates the brightness value (V) from the color chrominance, which can be further divided into hue (H) and saturation (S). Because these elements in HSV have a relatively weak correlation with each other, it is highly suitable for use in feature color extraction. In comparison to RGB, one of the advantages of the HSV color space is its weak inter-element correlation, making it easy to control. In applications involving color recognition, we can convert the detected images from the RGB color space to the HSV color space with Equation (1).

$$\begin{gathered} H = \cos^{-1}\left(\frac{\frac{1}{2}[(R-G)+(R-B)]}{\left[(R-G)^2+(R-B)(G-B)^2\right]}\right) \\ S = 1 - \frac{3[\min(R,G,B)]}{R+G+B} \\ V = \frac{\max(R,G,B)}{255} \end{gathered} \tag{1}$$

2.2.2. Extracting Feature Colors of Image

The feature color extraction process in [25] involves first segmenting the elements of an image's HSV color space, followed by the conversion of each element (H, S, V) into a histogram of oriented gradient (HOG). Since the HOG divides each element into several element intervals, the segmentation proportions for each element can be determined. Then, selecting the interval with the highest proportion for each element, we can obtain their respective numerical values (H, S, V). These values represent the HSV feature colors for the image.

2.2.3. Transformation of Color Space

After experimenting, it has been observed that certain issues exist when directly calculating color distances in the HSV color space. Specifically, when the saturation (S) is low, it often leads to the k-nearest neighbors (KNN) [26] decision result being mistakenly classified as gray, regardless of how the hue (H) changes. To address this, the extracted feature colors in HSV are transformed into the RGB color space using Equation (2) [25]. This transformation involves mapping the hue (h) range to h_i, and calculating variables p, q, t based on the hue (h_i) range to determine which combination of RGB attributes (p, q, t, v) applies. The calculated RGB values (r_0, g_0, b_0) are then subjected to Euclidean distance computation [27] against pre-established RGB color table values (r_1, g_1, b_1) to determine the color distance (d), as illustrated in Equation (3). Subsequently, the KNN algorithm is employed to identify the color of the clothing based on this computed distance.

$$\begin{aligned} h_i &= \left\lfloor \frac{h}{60} \right\rfloor \\ f &= \frac{h}{60} - h_i \\ p &= v \times (1 - s) \\ q &= v \times (1 - f \times s) \\ t &= v \times (1 - (1 - f) \times s) \\ (r, g, b) &= \begin{cases} (v, t, p), if\ h_i = 0 \\ (q, v, p), if\ h_i = 1 \\ (p, v, t), if\ h_i = 2 \\ (p, q, v), if\ h_i = 3 \\ (t, p, v), if\ h_i = 4 \\ (v, p, q), if\ h_i = 5 \end{cases} \end{aligned} \tag{2}$$

$$d = \sqrt{(r_1 - r_0)^2 + (g_1 - g_0)^2 + (b_1 - b_0)^2}, \tag{3}$$

2.2.4. K-Nearest Neighbors (KNN) Color Classification

K-nearest neighbors (KNN) [26] is a fundamental classification and regression algorithm. After obtaining the HSV feature colors of an image and calculating the color distances using Equation (3), these distances are compared to a pre-established RGB color table. After sorting the color distances for each color, K colors with the closest distances are then selected. Followed by a voting process among neighboring colors, the color with the most votes is determined as the final color result selected by the KNN algorithm.

2.2.5. UAV Systems for Human Detection

The work in [28] proposes an approach utilizing an automated human detection system on UAVs to identify human bodies, discussing the hardware configuration of UAVs and real-time human recognition capabilities. Ref. [29] presents a comprehensive human activity recognition algorithm, where the UAV first identifies whether the object is a person and subsequently recognizes various human activities, such as throwing, walking, and digging. Additionally, the study introduces various image stabilization techniques. The research of [15] focuses on achieving human body recognition using a CNN. Due to the difficulty in acquiring datasets, data augmentation is employed to enhance the training outcomes. The study compares the training outcomes using various architectures and outlines the algorithm's path planning as a spiral search. The focus of the study in [30] lies in the application of UAVs for commercial transportation, aiming to achieve successful human body recognition using UAVs. The research encompasses the design of five distinct scenarios, revealing that the distance variation between the UAV and the human body has a more significant impact on the recognition success compared to the quality of the camera. In the context of search and rescue operations for swimmers, ref. [31] proposes a methodology that integrates global navigation satellite system (GNSS) techniques with computer vision algorithms to locate individuals in distress. Refs. [32,33] primarily focus on the training of human detection models. Ref. [32] introduces a modified YOLOv8 architecture by incorporating the SC3T module into the final layer and training the model using images captured from a UAV perspective. The emphasis of the study lies in the recognition performance. The experimental results are evaluated using confusion matrices and the mean average precision. The findings reveal that, across the precision rate, recall rate, and mAP, the modified YOLOv8 outperforms both the original YOLOv5 and YOLOv8 models. Ref. [33] primarily utilizes YOLOv5 for human detection and further employs a Haar cascade classifier to identify specific body parts (head, upper body, lower body). The final results indicate that YOLOv5 achieves 98% average precision (AP), while the Haar cascade classifier attains approximately 78% AP. Table 1 presents a comparison of related studies on human detection using UAVs. It can be found that most of the related methods focus on how to improve the human body recognition model, without considering how to speed up the search in order to shorten the search time and search efficiency, which is the aim of this study. Hence, the integrated YOLOv5 and HWF framework is proposed here to obtain an efficient UAV searching system by combining the hierarchical human-weight-first (HWF) path planning algorithm with the results of human body recognition from the existing YOLOv5 model and the clothing/pant colors from KNN color recognition.

Table 1. Comparison of related studies of UAV human detection.

	Human Body Recognition Model	**Dataset Used**	**Recognition of Human Clothing Types and Colors**	**Segmentation of the Search Area**	**Dynamic Route Planning for Search**	**Integration of Human Body and Clothing/Pant Color Recognition with Dynamic Route Planning**
[28]	Motion detection outputs a score of human confidence	No	No	No	No	No
[29]	CNN	UCF-ARG dataset	No, proposes human activity classification algorithm	No	No	No
[15]	CNN	Self-developed captured dataset	No	No	No, spiral search	No
[30]	DNN with MobileNet V2 SSDLite	COCO dataset	No	No	Yes, estimates the person and moves in his direction with GPS	
[31]	CNN with Tiny YOLOv3	COCO dataset + self-developed swimmers dataset	No	No	No	No
[32]	CNN with modified YOLOv8	Self-developed UAV view real-world dataset	No	No	No	No
[33]	CNN with YOLOv5 and Haar Cascade classifier	VisDrone dataset + COC0128 dataset	No, proposes a human body region classification algorithm	No	No	No
HWF	CNN with YOLOv5	VisDrone dataset + self-developed drone-clothing dataset	Yes, uses KNN color recognition	Yes	Yes, proposes the hierarchical human-weight-first (HWF) path planning algorithm	Yes, Proposes the integrated YOLOv5 and HWF framework

3. System Architecture and Algorithms

3.1. System Architecture

As depicted in Figure 4, the system architecture is divided into three main components: the UAV equipped with the Jetson Nano [34], the server side, and the client side, where the Jetson Nano handles the UAV's flight commands and mobile network communication tasks. The search begins with the user inputting the location and the block size of the search area, as well as four target searching criteria, into the server.

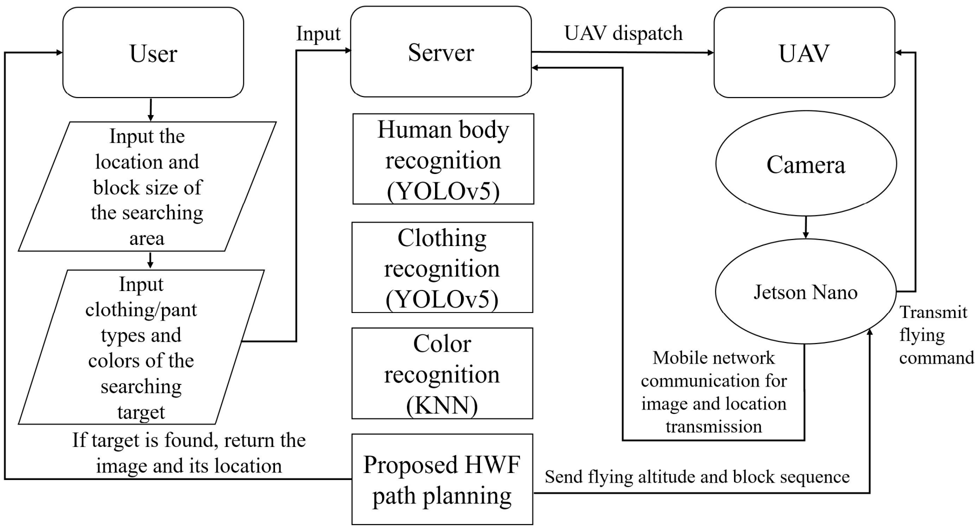

User
Input the location and block size of the searching area
Input clothing/pant types and colors of the searching target
Input
Server
Human body recognition (YOLOv5)
Clothing recognition (YOLOv5)
Color recognition (KNN)
Proposed HWF path planning
UAV dispatch
UAV
Camera
Jetson Nano
Transmit flying command
Mobile network communication for image and location transmission
If target is found, return the image and its location
Send flying altitude and block sequence

Figure 4. System architecture and flowchart of this study.

The four target searching conditions are shown below:

- Clothing type;
- Pant type;
- Clothing color;
- Pant color.

After the inputs, the server establishes a communication link with the Jetson Nano on the UAV via a mobile network connection, and the UAV is then dispatched from the starting point to the search area to initialize the search operation. Using a USB camera mounted on the UAV, aerial images are captured for the search area and blocks. All captured images and their shooting locations are sent back to the server, which benefits from its enhanced computational ability to undertake more intricate operations, including human body recognition by YOLOv5, clothing recognition by YOLOv5, and color identification by the KNN algorithm. This setup harnesses the advantages of cloud computing [35]. During the UAV's searching process according to the hierarchical human-weight-first (HWF) path planning algorithm, if the server identifies a search target that meets the search criteria, it notifies the user and the Jetson Nano on the UAV. This instruction prompts the UAV to return to the starting point, concluding its search mission.

3.2. Search Algorithm

3.2.1. Hierarchical Flight Altitudes for the UAV

In this study, the UAV's flight altitude is divided into two types: the initial altitude h_0 to capture the image of the whole search area and altitude h_1 to capture an image of a single block to achieve the optimal object recognition. The server conducts path planning at initial altitude h_0 and then directs the UAV to fly at the optimal object recognition altitude h_1 to traverse the blocks. According to Equation (4), the server calculates altitude h_0 by utilizing the side length of the search area, i.e., $\sqrt{area}$, and $tan\theta$, where θ is half of the field of view (FOV) of the UAV's camera. This calculation determines the UAV height h_0 at which the image captured by the UAV's camera with this field of view can cover the entire search area.

$$h_i = \frac{\sqrt{area}}{2tan\theta} \tag{4}$$

Subsequently, the optimal object recognition altitude h_1 is determined by calculation based on the predefined side length n of the block. At height h_1, the UAV has a clearer perspective to achieve better recognition results, enhancing the accuracy in identifying the search target. When the block size ($n \times n$) remains constant, the number of total blocks increases as the search area becomes larger. Conversely, as the size of the search area decreases, fewer blocks are segmented, as illustrated in Figure 5. After calculating the individual weight values for all blocks at initial altitude h_0, the human-weight-first (HWF) algorithm is employed to plan the flight path at the block level. Once the flight path is planned, the UAV descends from initial altitude h_0 to altitude h_1 to start the block search. During the search path of the block level, if the server identifies an object from the image captured by the UAV at altitude h_1 with a weight score exceeding the predefined threshold for the search target, it is considered that the target has been found. In this case, the server sends the target's location and the corresponding captured image back to the user and this concludes the UAV's search mission.

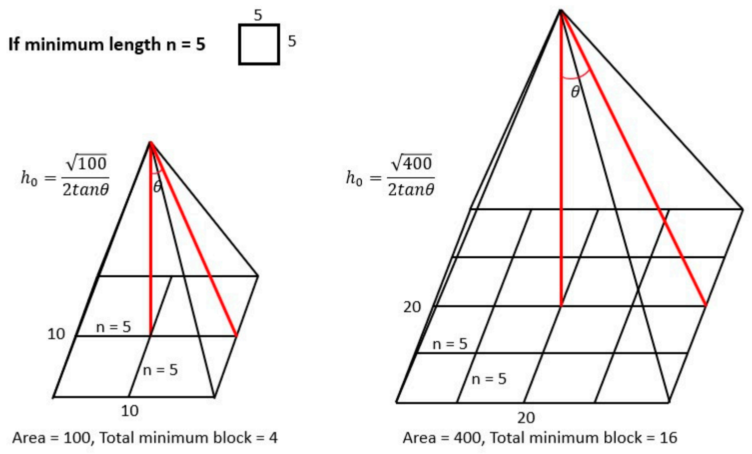

If minimum length n = 5
5
5
$h_0 = \frac{\sqrt{100}}{2tan\theta}$
θ
10
n = 5
n = 5
10
Area = 100, Total minimum block = 4
$h_0 = \frac{\sqrt{400}}{2tan\theta}$
θ
20
n = 5
n = 5
20
Area = 400, Total minimum block = 16

Figure 5. The number of blocks versus the area size.

3.2.2. Block Weight in the Search Area

The weight within each block is determined by the accuracy value of the recognized person's body (AP_{ha}), the accuracy values of the clothing and pant types (AP_{ca}, AP_{pa}), the recognized clothing and pant types (C_{rct}, C_{rpt}), the search clothing and pant types (C_{sct}, C_{spt}), the recognized clothing and pant colors (C_{rcc}, C_{rpc}), and the search clothing and pant colors (C_{scc}, C_{spc}) in a fuzzy manner. This calculation results in the weight value, i.e., human_weight, for the recognized person. The block weight, i.e., block_weight, is defined as the highest weight value among all recognized persons in a block. By sorting the block weight values in the whole search area, it is possible to determine which block is most likely to contain the search target.

3.2.3. Hierarchical Human-Weight-First (HWF) Path Planning Algorithm

Traditional research such as [35] proposes the exhaustive approach to generate all possible paths, select the shortest path in terms of the path length, and finally traverse the blocks in descending order of the block weight. However, the exhaustive algorithm exhibits exponential time complexity as the size of the search area increases. Therefore, in this study, the human-weight-first (HWF) algorithm is designed as the path planning algorithm for the UAV. The HWF algorithm makes the optimal choice at each step in the current state, as in [36], aiming to achieve an overall result that is as optimal as possible.

The flow of the HWF algorithm is listed below and illustrated in Figure 6.

(a) The UAV flies from the center point of the search area to an altitude of h_0 to begin recognition, which is shown in Figure 6a. Block weights are calculated using the image captured by the UAV at altitude h_0.
(b) The HWF algorithm selects the block with the highest block weight as the starting point for the block search and guides the UAV to descend to the center of the block at

altitude h_1, which is shown in Figure 6b. The UAV then captures images of the block and sends them to the server for further recognition.

(c) If no search target is found in this block, HWF instructs the UAV to traverse to the block with the next highest block weight until the search target is found, i.e., the block weight exceeds the search target threshold, or all blocks with nonzero block weights have been visited, as shown in Figure 6c.

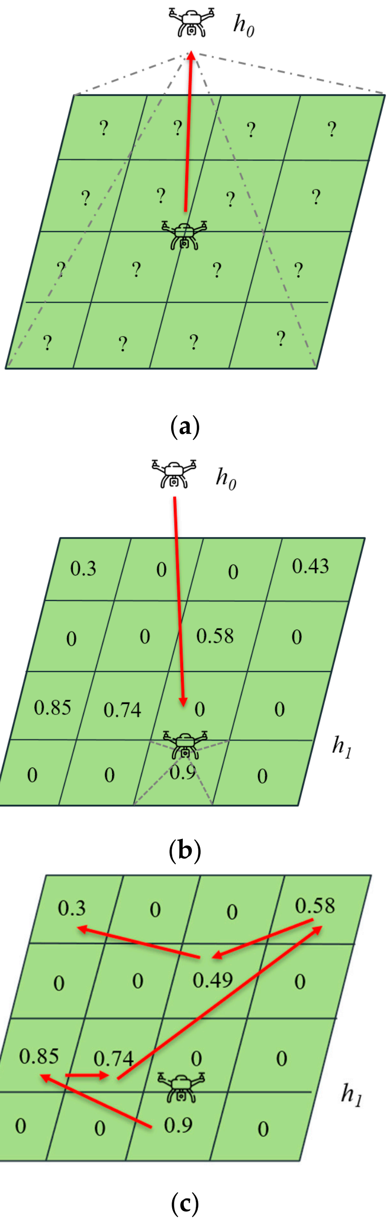


Figure 6. The complete planned path by HWF. (**a**) The UAV flies from the center point of the search area to an altitude of h_0 to begin recognition. (**b**) The HWF algorithm selects the block with the highest block weight as the starting point for the block search and guides the UAV to descend to the center of the block at altitude h_1. (**c**) The red lines show the complete block traversal order.

As depicted in Figure 6, the value in a block represents its block weight, which is used to prioritize the block search order, increasing the speed when identifying the search target. Hence, the complete block traversal order in Figure 6c is as follows: block [0.9] $\rightarrow$ block [0.85] $\rightarrow$ block [0.74] $\rightarrow$ block [0.49] $\rightarrow$ block [0.58] $\rightarrow$ block [0.3]. It should be noted that the proposed integrated YOLOv5 and HWF approach results in a high probability of recognizing the search target in the first few blocks, which significantly reduces the UAV search time, traversal distance, and power consumption.

3.2.4. Convenient Visit Algorithms Based on HWF

In Figure 7, the black values represent the block weights calculated at altitude h_0, while the blue values represent the weight scores calculated at altitude h_1. Assume that the target threshold is set as 80. Since the weights at both h_0 and h_1 altitudes do not exceed the target threshold, the HWF path planning algorithm performs a complete search over all blocks with nonzero weights, i.e., from the block with weight 76.0 to that with weight 5.1. This means that the HWF algorithm may lead to a UAV flight path that passes through some intermediate blocks several times, which increases the search path length, search delay, and UAV power consumption accordingly. This situation intensifies significantly when the search area expands. To address this issue, we design two variants of the HWF algorithm to search these intermediate blocks when it is convenient, if certain conditions are met, thus addressing the aforementioned problems.

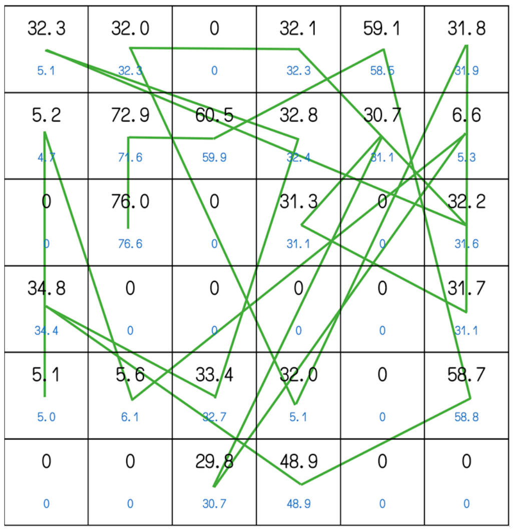


Figure 7. The complete HWF search path under a 6 × 6 search area. The black values represent the block weights calculated at altitude h_0, while the blue values represent the weight scores calculated at altitude h_1.

A threshold for convenient visits, which is called the convenient threshold, is defined in advance. If the UAV passes through an intermediate block with a weight exceeding the convenient threshold along the HWF flight path from the current block to the next one, it will be guided to the center of this block for the convenient visit. This approach reduces the likelihood of unnecessary, repeated UAV flights among blocks. The value of the convenient threshold should be strategically defined to strike a balance. If it is set too high, the HWF variant could result in a UAV traversal path similar to that of the original HWF. Conversely, if it is set too low, the HWF variant could lead to excessive detours, which may diminish the HWF search efficiency.

We propose two convenient visit algorithms based on the HWF approach as follows.

HWFR-S: A static and fixed convenient visit threshold is set. If there is a block along the HWF flight path that exceeds this convenient visit threshold, it will be visited by the UAV accordingly. In Figure 8, the red path represents the original HWF route, while the blue path represents the HWFR-S traversal route. HWFR-S sets a fixed convenient visit threshold of 0.7. Compared to the HWF algorithm, which guides the UAV from the block with a weight of 0.9 to the block with a weight of 0.85, HWFR-S reroutes to the intermediate block with a weight of 0.74 along the way. However, it does not choose to reroute to the block with a weight of 0.49 when the UAV flies from the block with a weight of 0.85 to the block with a weight of 0.58, because the weight of 0.49 does not exceed the convenient visit threshold of 0.7.

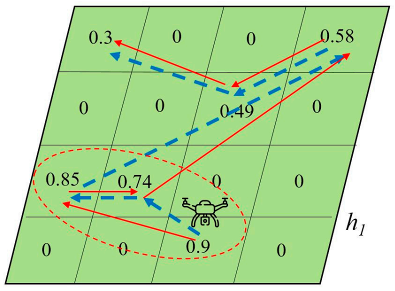


Figure 8. The original path (red line) of HWF and the modified path (blue line) of HWFR-S search algorithms.

HWFR-D: This algorithm proposes to use a fixed weight difference to calculate a dynamic convenient visit threshold for the next block. Its value is equal to the difference between the weight of the next block and the fixed weight difference, which means that the current visit threshold is dynamically adjusted by subtracting the weight difference from the next block's weight value. If there is an intermediate block with a weight no less than the current convenient visit threshold, the UAV will take a detour to visit this intermediate block and then return to visit the original next block, instead of visiting the original next block directly.

In Figure 9, the red path represents the original HWF route, while the purple path represents the HWFR-D route when HWFR-D sets the fixed weight difference as 0.1. With the original HWF algorithm, the UAV first visits the block with a weight of 0.74, followed by the block with a weight of 0.58, and finally the block with a weight of 0.49. In contrast, when the UAV flies from the block with a weight of 0.74 to the block with a weight of 0.58, HWFR-D dynamically calculates its convenient visit threshold as 0.48, which is equal to 0.58 − 0.1. Then, HWFR-D chooses to conveniently visit the intermediate block with a weight of 0.49, because the weight of 0.49 is greater than the current convenient visit threshold of 0.48. In contrast, the intermediate block with the weight of 0.74 is not selected for the convenient visit in the HWFR-D route when the UAV flies from the block with a weight of 0.9 to the block with a weight of 0.85, since the weight of 0.74 is less than the current convenient visit threshold, i.e., 0.75 = 0.85 − 0.1.

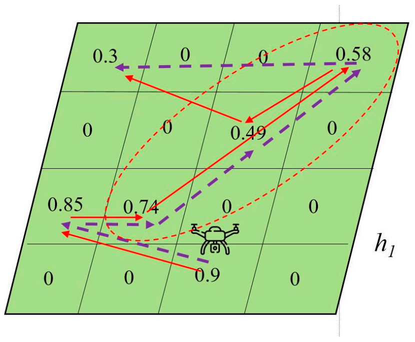
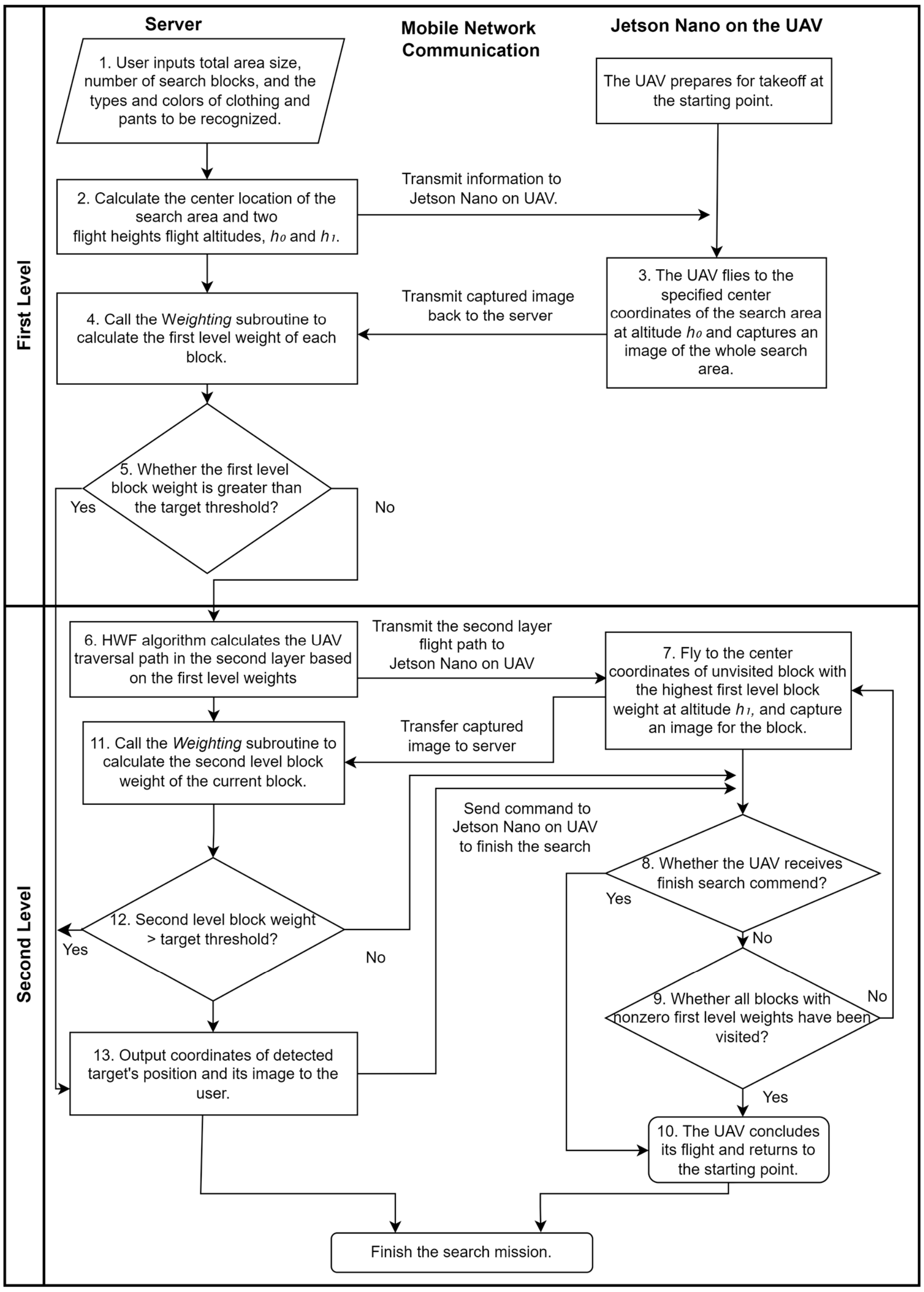


Figure 9. The original path (red line) of HWF and the modified path (purple dotted line) of HWFR-D search algorithms.

3.2.5. Flow of the Integrated YOLOv5 and HWF Framework

The flow of the integrated YOLOv5 and HWF framework, which tightly integrates the searching UAV equipped with the Jetson Nano and the cloud server at altitudes h_0 and h_1 in the search and rescue process, is illustrated in Figure 10.



First Level
Second Level
Server
Mobile Network Communication
Jetson Nano on the UAV
1. User inputs total area size, number of search blocks, and the types and colors of clothing and pants to be recognized.
The UAV prepares for takeoff at the starting point.
2. Calculate the center location of the search area and two flight heights flight altitudes, h0 and h1.
Transmit information to Jetson Nano on UAV.
3. The UAV flies to the specified center coordinates of the search area at altitude h0 and captures an image of the whole search area.
Transmit captured image back to the server
4. Call the Weighting subroutine to calculate the first level weight of each block.
5. Whether the first level block weight is greater than the target threshold?
Yes
No
6. HWF algorithm calculates the UAV traversal path in the second layer based on the first level weights
Transmit the second layer flight path to Jetson Nano on UAV
7. Fly to the center coordinates of unvisited block with the highest first level block weight at altitude h1, and capture an image for the block.
Transfer captured image to server
11. Call the Weighting subroutine to calculate the second level block weight of the current block.
Send command to Jetson Nano on UAV to finish the search
8. Whether the UAV receives finish search commend?
Yes
No
12. Second level block weight > target threshold?
Yes
No
9. Whether all blocks with nonzero first level weights have been visited?
No
Yes
13. Output coordinates of detected target's position and its image to the user.
10. The UAV concludes its flight and returns to the starting point.
Finish the search mission.

Figure 10. Flowchart of the integrated YOLOv5 and HWF framework.

1. The user inputs the values of six parameters, i.e., the total search area size, the number of search blocks, and the features of the search target, including the types and colors of their clothing and pants, at the server side. At the same time, the UAV prepares for takeoff at the origin point.
2. The server calculates the center location of the search area and two flight heights, i.e., h_0 and h_1, for the searching UAV and sends this information to the Jetson Nano on the UAV through the mobile network communication.
3. The UAV flies to the specified center coordinates of the search area at altitude h_0 and captures an image of the whole search area, which is then transmitted back to the server.
4. The server executes the weighting subroutine to calculate the first-level weights of all blocks within the total search area.

5. If the first-level weight of a particular block is greater than the search target threshold, the system proceeds to step 13; otherwise, it proceeds to step 6.
6. The system plans the second-level traversal path for the blocks at altitude h_1, based on the first-level block weights, using the HWF algorithm. Then, the server transmits the planned path for the second layer to the Jetson Nano on the UAV.
7. The UAV flies to the center coordinates of the unvisited block with the highest first-level block weight at altitude h_1, according to the planned path, and captures an image of the block. This block image is then transmitted back to the server.
8. If the UAV receives a command to finish the search, it proceeds to step 10; otherwise, it proceeds to step 9.
9. If all blocks with nonzero first-level weights have been visited by the UAV, the system proceeds to step 10; otherwise, it proceeds to step 7.
10. The UAV concludes its flight and returns to the starting point.
11. Whenever the server receives a block image transmitted by the UAV at step 7, it runs the weighting subroutine again to calculate the second-level block weight of the current search block.
12. If the second-level block weight is greater than the search target threshold, the system proceeds to step 13; otherwise, it returns to step 8.
13. The system outputs the coordinates of the detected target's position along with its image to the user, which indicates that the search target has been found. The server then sends a command to the Jetson Nano on the UAV to finish the search mission.

3.2.6. Weighting Subroutine Flowchart

1. As shown in Figure 11, the server first sets the initial values of the total weight value (W), human weights, and block weight as 0. It then executes YOLOv5 for human detection on UAV-captured images.
2. If a human body is detected, the server extracts the human body image with its bounding boxes and proceeds to use YOLOv5 for clothing recognition at step 3.
3. The server executes YOLOv5 for clothing recognition on the extracted human body image.
4. If the clothing is recognized and the recognized clothing type (C_{rct}) matches the search clothing type (C_{sct}), the total weight value (W) is incremented by the minimum value between the clothing accuracy (AP_{ca}) and the custom clothing fuzzy threshold of 0.3. Then, it proceeds to step 5. Otherwise, the system proceeds to step 6.
5. The server performs the KNN color recognition on the recognized clothing.
6. If the recognized clothing color (C_{rcc}) matches the search clothing color (C_{scc}), the total weight value (W) is incremented by the minimum value between the KNN color percentage and the custom clothing color fuzzy threshold of 0.2. Then, it proceeds to step 7.
7. If the pants are recognized and the recognized pant type (C_{rpt}) matches the search pant type (C_{spt}), the total weight value (W) is incremented by the minimum value between the pant accuracy (AP_{pa}) and the custom pant fuzzy threshold of 0.3. The system proceeds to step 8; otherwise, it proceeds to step 9.
8. The server performs KNN color recognition on the recognized pants.
9. If the recognized pant color (C_{rpc}) matches the search pant color (C_{spc}), the total weight value (W) is incremented by the minimum value between the KNN color percentage and the custom pant color fuzzy threshold of 0.2.
10. The weight score of a person (human_weight) is calculated by the weighted function of the human accuracy value (AP_{ha}) and the total weight value (W), with the coefficient of 0.1 and 0.9, respectively. The maximum person weight value within a block is defined as the block_weight.

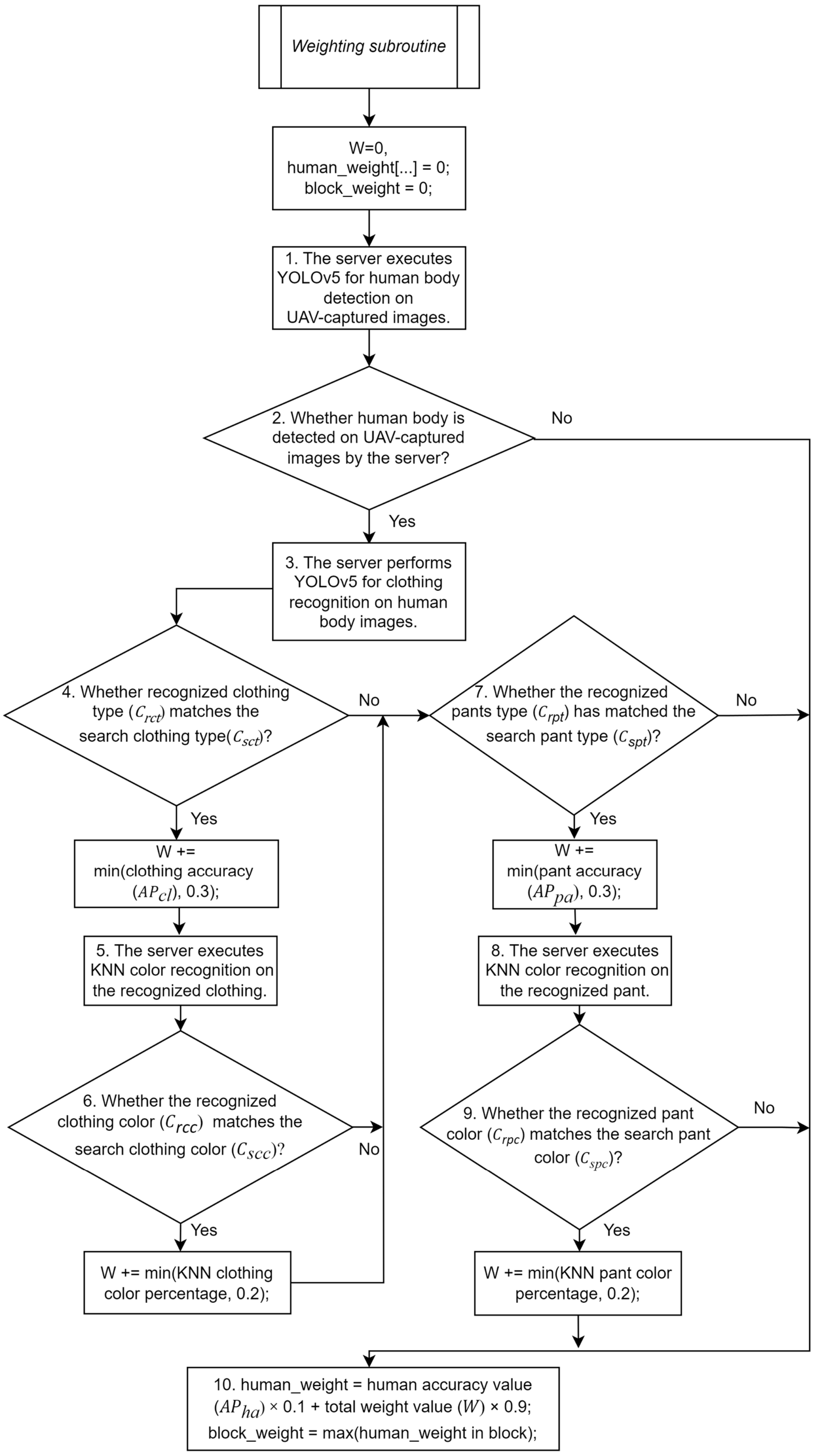

Weighting subroutine
W=0, human_weight[...] = 0; block_weight = 0;
1. The server executes YOLOv5 for human body detection on UAV-captured images.
2. Whether human body is detected on UAV-captured images by the server?
No
Yes
3. The server performs YOLOv5 for clothing recognition on human body images.
4. Whether recognized clothing type (C_{rct}) matches the search clothing type(C_{sct})?
No
Yes
W += min(clothing accuracy (AP_{cl}), 0.3);
5. The server executes KNN color recognition on the recognized clothing.
6. Whether the recognized clothing color (C_{rcc}) matches the search clothing color (C_{scc})?
No
Yes
W += min(KNN clothing color percentage, 0.2);
7. Whether the recognized pants type (C_{rpt}) has matched the search pant type (C_{spt})?
No
Yes
W += min(pant accuracy (AP_{pa}), 0.3);
8. The server executes KNN color recognition on the recognized pant.
9. Whether the recognized pant color (C_{rpc}) matches the search pant color (C_{spc})?
No
Yes
W += min(KNN pant color percentage, 0.2);
10. human_weight = human accuracy value (AP_{ha}) × 0.1 + total weight value (W) × 0.9; block_weight = max(human_weight in block);

Figure 11. Flowchart of the weighting subroutine.

3.2.7. KNN Color Recognition Process

The flowchart of the KNN color recognition process is shown in Figure 12. Its details are listed below.

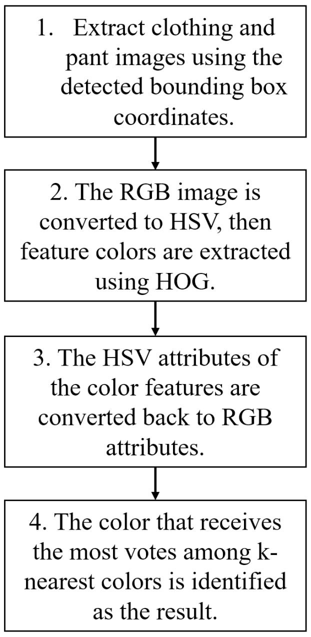

1. Extract clothing and pant images using the detected bounding box coordinates.
2. The RGB image is converted to HSV, then feature colors are extracted using HOG.
3. The HSV attributes of the color features are converted back to RGB attributes.
4. The color that receives the most votes among k-nearest colors is identified as the result.

Figure 12. Flowchart of the KNN color recognition process.

1. After correctly identifying the types of clothing and pants, the system proceeds to extract clothing and pant images using the detected bounding box coordinates. Subsequently, the system applies noise reduction techniques to the image, facilitating the extraction of feature colors.
2. The captured clothing and pant images are converted from the RGB color space to the HSV color space. Subsequently, OpenCV [37] is used to generate a color gradient direction histogram for the image. From this histogram, the algorithm selects the interval with the highest proportion, obtaining a new set of HSV values, which serves as a representation of the image's feature color.
3. The feature color representing the HSV color attributes is converted back to RGB color attributes.
4. The color distances between the image's feature color and the RGB color table established in this study are computed. Subsequently, these distances are arranged in order, and k-nearest colors are chosen by a voting process. The color that receives the most votes is identified as the result of the KNN color recognition process.

4. Simulation Results

4.1. YOLOv5 Image Recognition Model

In this study, we utilize YOLOv5 for human body and clothing/pant recognition. These recognitions are executed on a cloud server. The human body database utilized for training in this study is sourced from VisDrone2019 [38], which is a dataset containing human-related images captured from the perspective of UAVs. It comprises a total of 7400 images, with 5600 images for training, 530 for validation, and 1270 for testing. The clothing/pant dataset is constructed by us and named the drone-clothing dataset, which contains a total of 5000 images, with 4000 images used for training, 500 for validation, and 500 for testing. It is also a dataset captured from the perspective of a UAV. The clothing and pants are categorized into four distinct types: short sleeves, long sleeves, short pants, and long pants. Both models are trained for 150,000 iterations on a server over the duration of one month.

The precision and mean average precision (mAP) values of the VisDrone2019 model are illustrated in Figures 13 and 14. Precision measures whether the recognized target is indeed the intended target when it is detected, while mAP measures the confidence of the recognition when a target is detected and incorporates a trade-off between precision and recall. Figure 15 presents the original images and recognition results of the VisDrone2019 model. Because most original images in the VisDrone2019 model are shot at long distances, as shown in Figure 15, which results in smaller and more complex objects, the recognized results are more likely to output incorrect results. Hence, the precision and mAP values of the training set in the VisDrone2019 model, shown in Figures 13 and 14, are relatively stable after epoch 160 and reach 0.6773 and 0.5020 at epoch 460, respectively. Table 2 lists the values of the three classification metrics, i.e., the precision, recall, and mAP values, of the training set and the testing set in the VisDrone2019 model at epoch 460. These values for the training set are not notably higher than those of the testing set, which means that human body recognition using the VisDrone2019 model shows no significant overfitting behavior in this study. On the other hand, the precision and mean average precision (mAP) values of the drone-clothing model are illustrated in Figures 16 and 17. Figure 18 presents the original images and recognition results of the drone-clothing model. As shown in Figure 18, because most original images in the drone-clothing model are shot at distances closer to the human body than those in the VisDrone2019 model, the recognition results for human clothing are more precise. Hence, the precision and mAP values of the training set in the drone-clothing model, shown in Figures 16 and 17, are relatively stable after epoch 40 and reach 0.9556 and 0.9540 at epoch 160, respectively. Consequently, the drone-clothing model converges faster and achieves higher classification metric values than the VisDrone2019 model does.

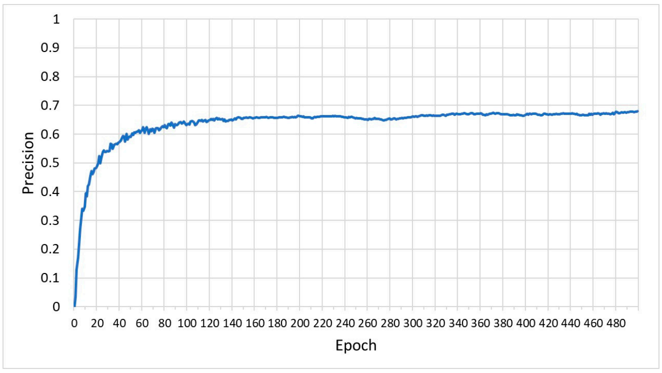


Figure 13. Precision values (blue line) of the VisDrone2019 model.

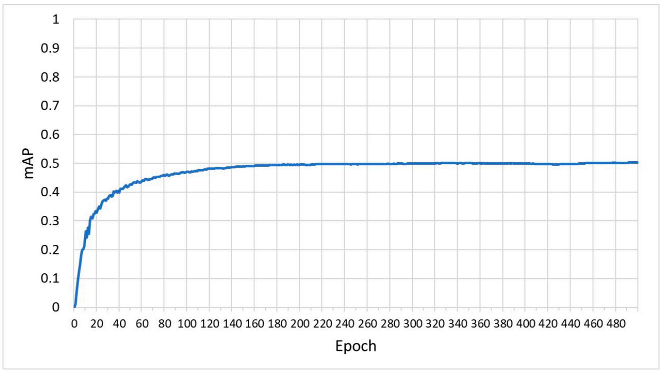


Figure 14. mAP values (blue line) of the VisDrone2019 model.

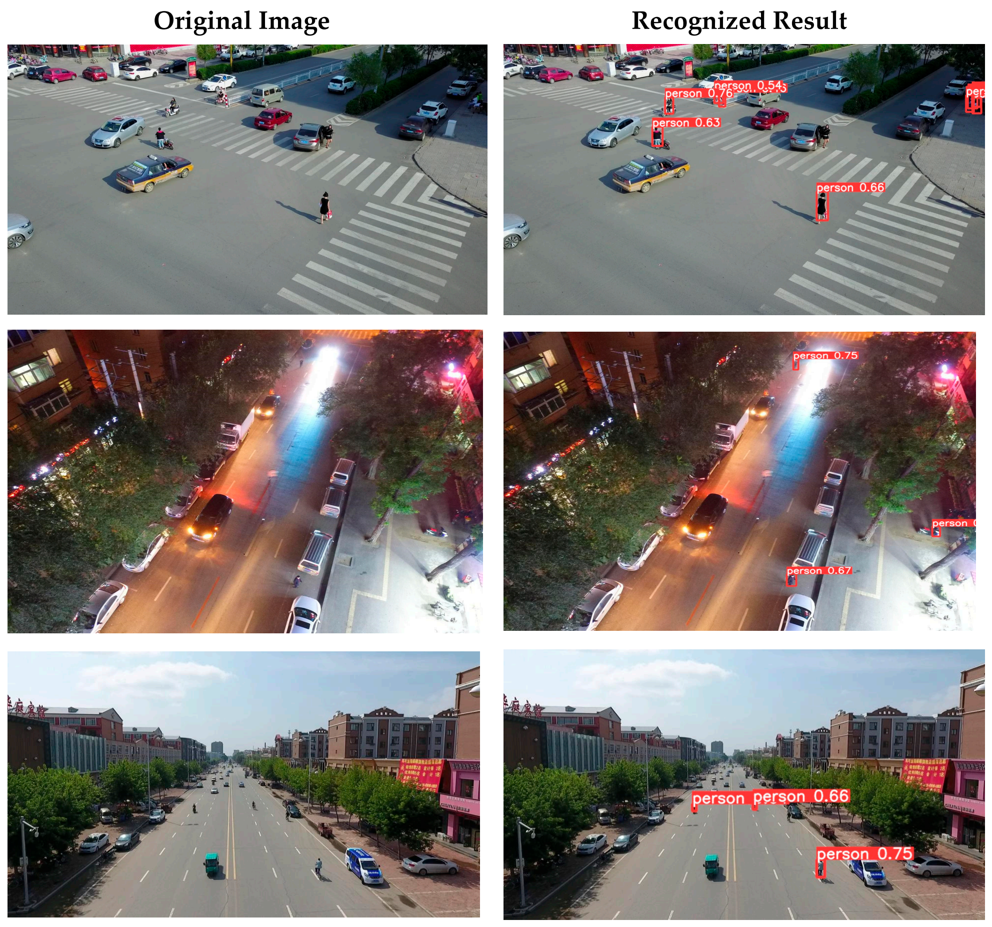


Figure 15. Original images and recognition results of the VisDrone2019 model.

Table 2. Values of classification metrics of the training set and the testing set in the VisDrone2019 model.

Classification Metric	**Training Set**	**Testing Set**
Precision	0.6773	0.6330
Recall	0.4887	0.4074
mAP	0.5020	0.3970

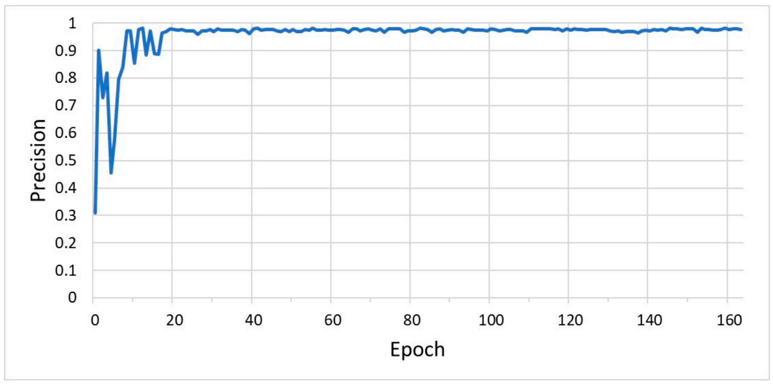


Figure 16. Precision values (blue line) of the drone-clothing model.

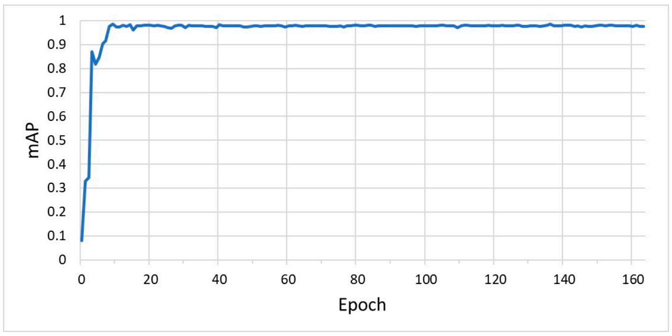


Figure 17. mAP values (blue line) of the drone-clothing model.

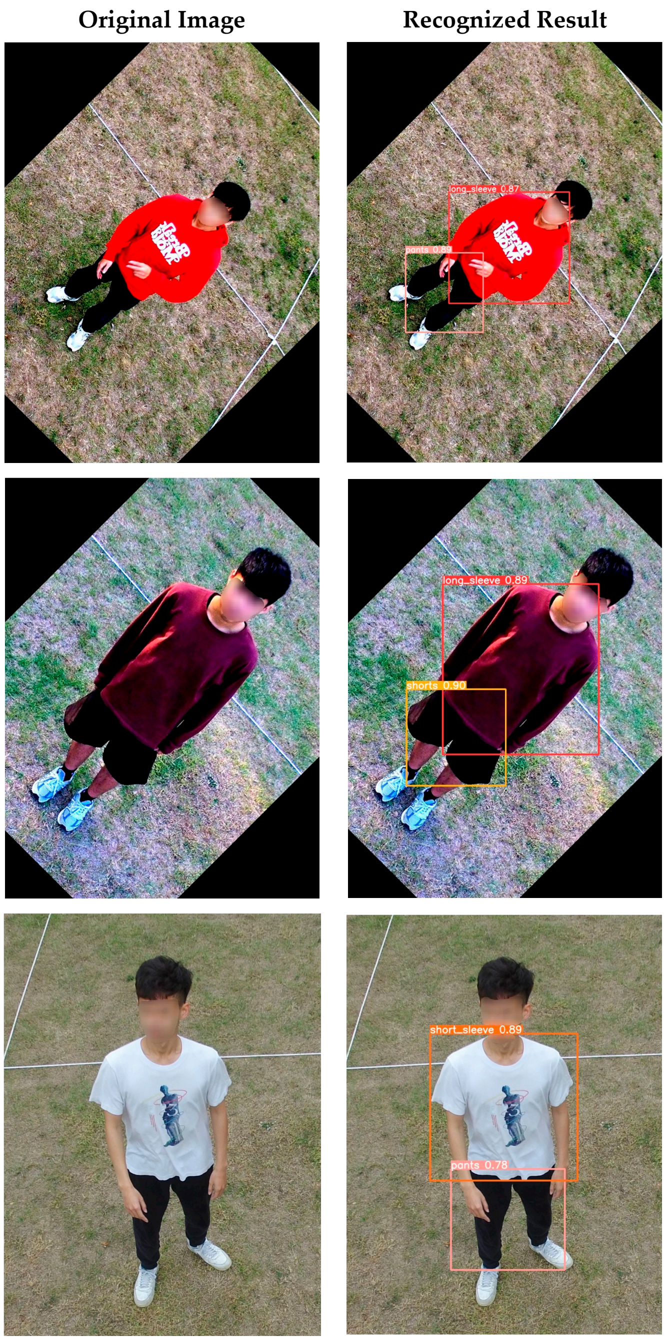


Figure 18. Original images and recognition results of the drone-clothing model.

4.2. Simulation Environment for Search Algorithms

In the simulation environment, Python is primarily used as the main programming language. The simulation area is partitioned into block sizes ranging from 3×3 to 10×10 and each block is assigned a weight. A total of 10,000 testing data are generated and each of them only gives a search target, which is randomly distributed within a specific block. We compare the proposed HWF, HWFR-S, and HWFR-D search algorithms with the sweep line and the spiral search algorithms in terms of four performance metrics, i.e., the average search path length, the average number of search blocks, the average search time, and the average search accuracy, required to locate the target location during the simulation of a UAV flight.

For each set of testing data, a single search target is randomly placed in a block within the $n \times n$ search area. Assume that the AP value of each search target in the captured image has different error ranges, depending on the UAV altitude. The features, i.e., the clothing type, pant type, clothing color, and pant color, of the search target are given different probabilities to change to different types and colors. The weighting subroutine calculates the weight value, i.e., human_weight, of each recognized person in every block based on the AP values of the four target features and determines whether the weight value of this person exceeds the search target weight threshold. The six simulation conditions are listed below. The simulation parameters are shown in Table 3.

Table 3. Simulation parameters.

Parameter	Value
Search area (m^2)	$\{(3n)^2 \mid n = 3, 4, \ldots, 10\}$
θ	40 degrees
Altitude h_i (m)	$\frac{\sqrt{area}}{2tan\theta}$
The percentage of error and probability variation at altitude h_0	10%, 20%, 30%
The percentage of error and probability variation at altitude h_1	5%, 10%, 15%
UAV velocity	20 km/h
UAV hovering and image capture time for a block	5 s
Search target threshold	0.7
Convenient visit threshold of HWFR-S	0.4, 0.5, 0.6
Weight difference of HWFR-D	0.1, 0.2, 0.3

1. The AP values for the search target's human body, types of clothing and pants, and colors of clothing and pants are randomly distributed between 0.9 and 0.99. The search target is randomly assigned to a specific block. In contrast, the AP values of the human body and types of clothing and pants for the person that is not the search target are randomly distributed between 0.01 and 0.99. The AP values of their clothing and pant colors are randomly set between 0.1 and 0.9.
2. Each block contains one to four persons within it, with the probability of 70%. It has a 30% probability of having no person in one block, which means that the weight of this block is set to zero accordingly.
3. At a higher altitude h_0, a larger error and probability variation of N% is applied to the given AP values and each feature of the person, respectively. Three different N% values, i.e., 10%, 20%, and 30%, are given to evaluate the performance metrics of these searching schemes.
4. At a lower altitude h_1, a smaller error and probability variation of 0.5 N% is applied to the given AP values and each feature of the person, respectively. Hence, three different 0.5 N% values, i.e., 5%, 10%, and 15%, are set accordingly.
5. Because most available USB cameras support resolutions such as 640×480 or 1280×720, we assume that the side length of each block is limited to 3 m, and

the area of each block is 9 square meters, such that the USB camera can capture vivid images for the whole search area and each block. Therefore, the total search area for an $n \times n$ block is $(3n)^2$ square meters.

6. As mentioned in Section 3.2.4, HWFR-S instructs the UAV to visit the block that exceeds the static and fixed convenient visit threshold. If the value of the convenient visit threshold is too small, the UAV has a higher probability of rerouting to a block without the search target, which increases the search path length and search time accordingly. In contrast, if the value of the convenient visit threshold is too large, the UAV may lose the opportunity to reroute to the block with the search target. Hence, the convenient visit threshold of HWFR-S is given intermediate values between (0, 1) as 0.4, 0.5, and 0.6 in this simulation. HWFR-D is given a fixed weight difference to calculate a dynamic convenient visit threshold by subtracting the weight difference from the next block's weight value. If there is an intermediate block with a weight no less than the current convenient visit threshold, the UAV will take a detour to visit this intermediate block. If the value of the weight difference is too large, which results in a smaller convenient visit threshold, the UAV suffers from a longer search path length and search time due to HWFR-S. Hence, the weight difference of HWFR-D is given lower values of 0.1, 0.2, and 0.3 in this simulation.

4.2.1. Average Search Path Length

As illustrated in Figure 19, the spiral and sweep search algorithms do not prioritize the searching order of blocks that may potentially contain the target. Under varying errors and probability variations at altitudes h_0 and h_1, as shown in Figure 19a–c, the average search path lengths of these two algorithms with 10,000 testing data are nearly identical. Furthermore, as the block size increases, the search path lengths of both algorithms grow linearly, resulting in the top two longest search path lengths among the five search algorithms considered. It should be noted that these two algorithms have the same search path lengths, as shown in Figure 19a–c, regardless of the set of N% and 0.5 N% error and probability variations at altitudes h_0 and h_1 given in the simulation, because they follow their fixed block search patterns, which are not dependent on the N value.

As the HWF algorithm and its variants prioritize the searching order of blocks that may potentially contain the target, they exhibit significant reductions in their search path lengths compared to the spiral and sweep algorithms. As mentioned above, the HWF path planning algorithm performs a complete search over all blocks with nonzero weights and it may lead to a UAV search path that passes through some intermediate blocks several times, especially when none of the block weights surpass the search target threshold. Hence, HWF can lead to an increase in the search path length.

The HWFR-S algorithm, due to its fixed convenient visit threshold, becomes more effective at reducing its search path length when more block weights exceed this threshold. If the threshold is set lower, the search path length of HWFR-S becomes shorter. Hence, HWFR-S with the convenient visit threshold of 0.4, denoted as HWFR-S (0.4) in Figure 19a–c, achieves shorter search path lengths than HWFR-S (0.5) and HWFR-S (0.6). On the other hand, when the weight difference is larger, which results in a lower convenient visit threshold, the variant of HWFR-D achieves shorter search path lengths. Hence, HWFR-D with the weight difference of 0.3, denoted as HWFR-D (0.3) in Figure 19a–c, achieves shorter search path lengths than HWFR-D (0.2) and HWFR-D (0.1). Hence, the HWFR-D algorithm has the capability to dynamically adjust the convenient visit threshold during the search process, making it the most successful algorithm to shorten the search path lengths. Moreover, if a larger error and probability variation on the AP value is given in the simulation, the difference between the calculated first-/second-level weight value (based on the AP values of the four target features of each recognized person in every block) and its correct value becomes larger. Hence, the HWF algorithm and its variants have a higher probability of deploying the UAV to an incorrect block to search for the search target, which results in a longer search path length. As shown in Figure 19a–c, the higher the N%

and 0.5 N% values are, the longer the search path lengths of the HWF algorithm and all its HWFR-S and HWFR-D variants. It should be noted that the HWFR-D algorithm with a larger error and probability variation outperforms HWFR-S and HWF significantly.

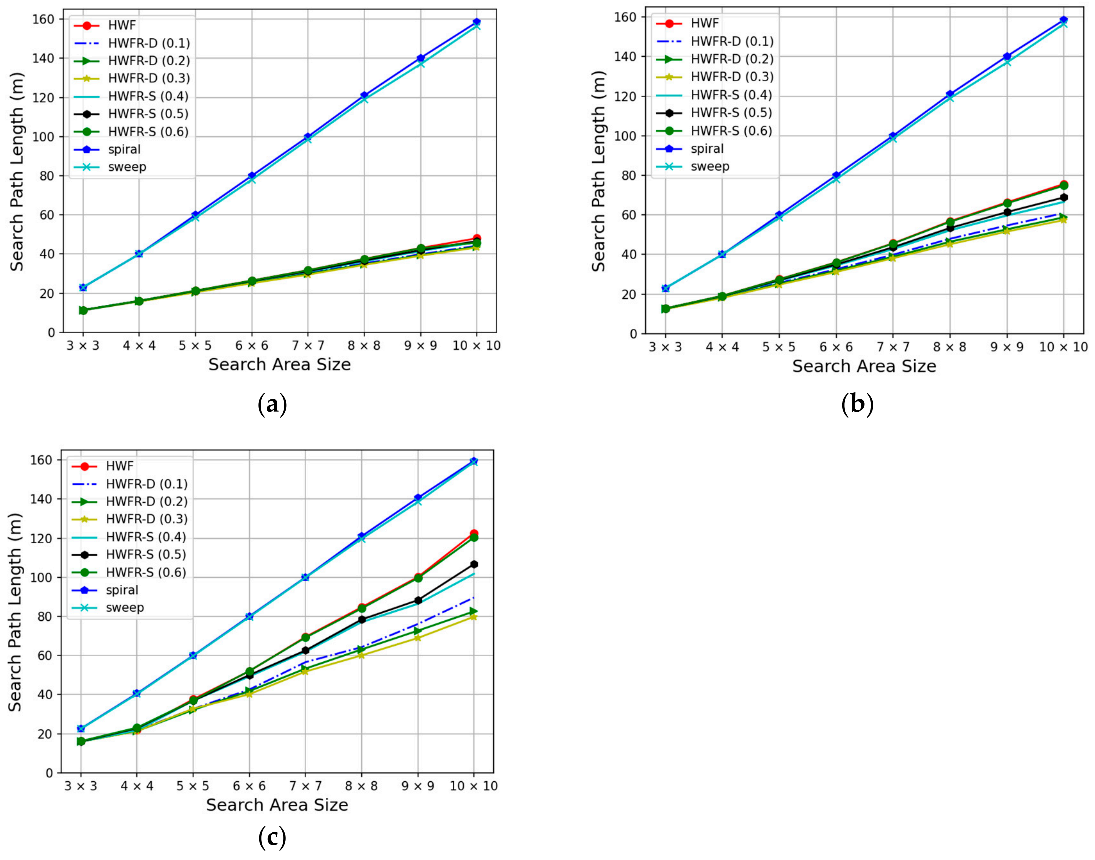


Figure 19. Average search path length vs. search area size for (**a**) 10%, 5%; (**b**) 20%, 10%; (**c**) 30%, 15% error and probability variations at altitudes h_0 and h_1.

4.2.2. Average Number of Search Blocks

As depicted in Figure 20a–c, both the spiral and sweep search algorithms do not prioritize the search order of blocks that may potentially contain the target. With fewer than 10,000 testing data, the average number of search blocks required before finding the search target for these two algorithms is similar. Moreover, as the search area size increases, the number of search blocks of both algorithms increases linearly, causing them to display the highest number of search blocks among the five algorithms considered. It should be noted that these two algorithms have the same number of search blocks, as shown in Figure 20a–c, regardless of the set of N% and 0.5 N% error and probability variations at altitudes h_0 and h_1 given in the simulation, because they follow their fixed block search patterns, which are not dependent on the N value.

The HWF, HWFR-S, and HWFR-D algorithms all share the characteristic of prioritizing the search order of blocks that is most likely to contain the search target. Since the first step in these algorithms is to select the block with the highest weight, the probability of finding the target at the first block is very high. This scenario results in the number of search blocks being equal to 1 for all three algorithms.

In the second scenario, when the HWF, HWFR-S, and HWFR-D algorithms encounter a situation wherein none of the block weights exceed the search target threshold during the search process, they will visit all blocks once and report that the search target cannot be found. In this case, the number of search blocks is equal to $n \times n$ for all three algorithms.

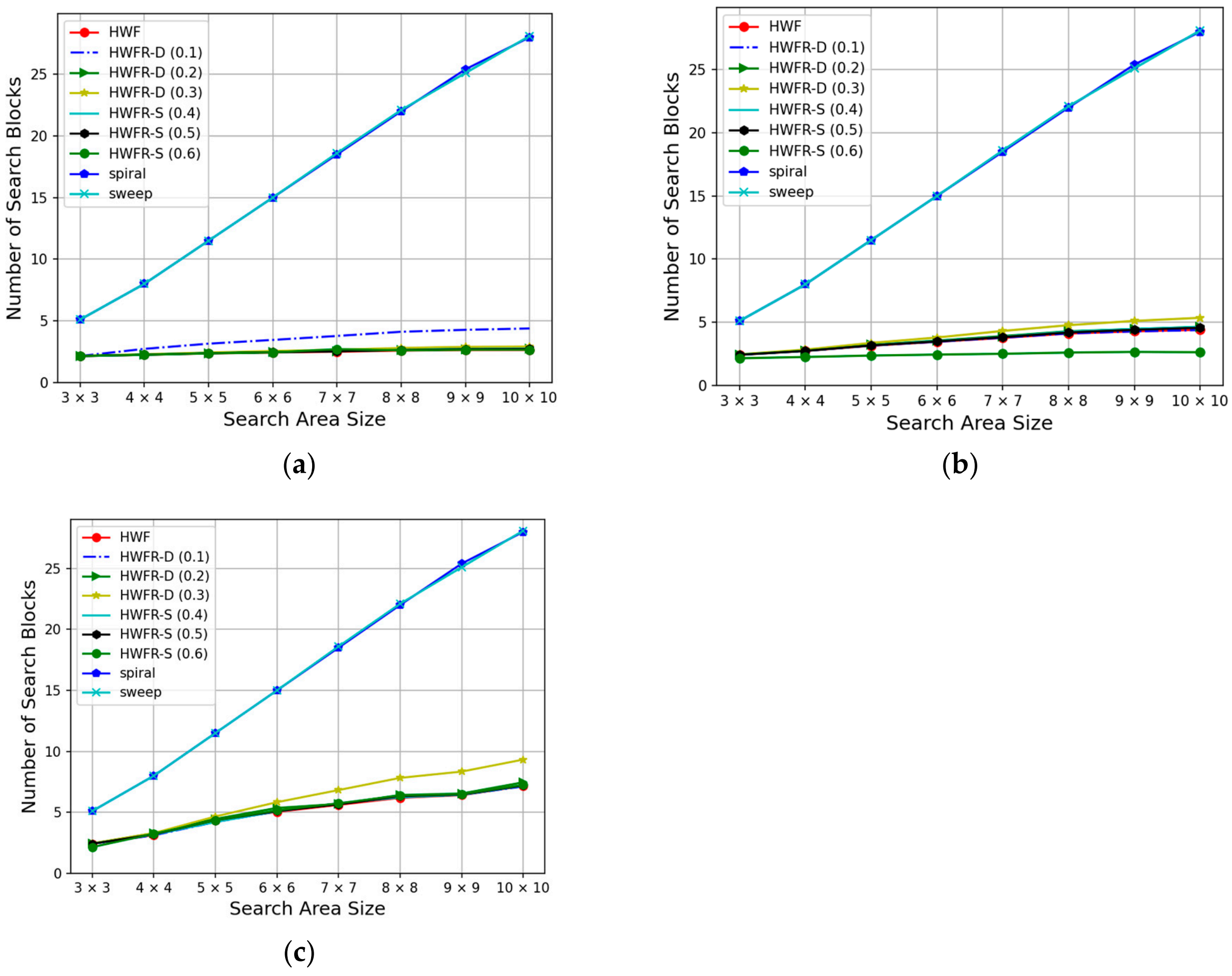
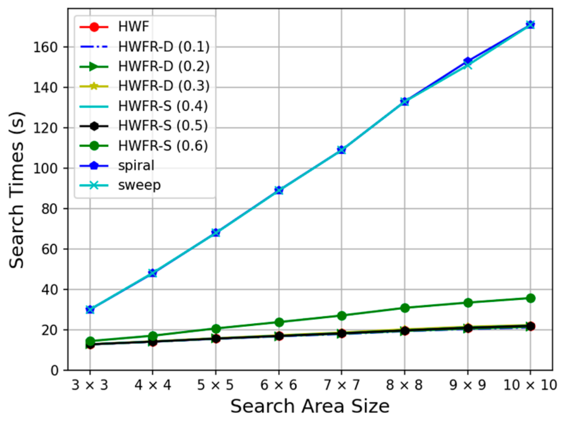


Figure 20. The average number of search blocks vs. search area size for (**a**) 10%, 5%; (**b**) 20%, 10%; (**c**) 30%, 15% error and probability variations at altitudes h_0 and h_1.

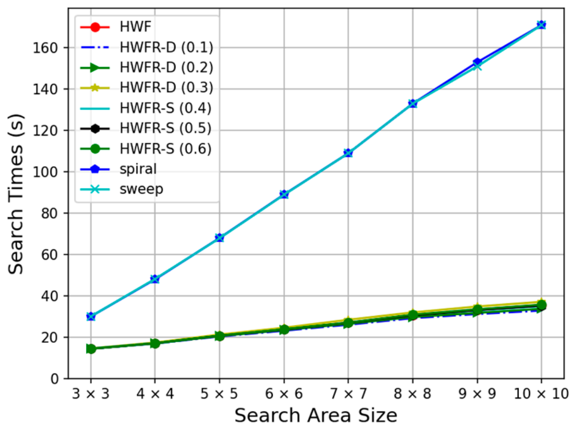
In the third scenario, the search target is found after the first step. HWFR-D exhibits an increase in the number of search blocks as its weight difference is larger. This means that HWFR-D with a larger weight difference will lower its convenient visit threshold and has a higher probability of rerouting to more intermediate blocks. In Figure 20a–c, the weight difference is set to 0.3, i.e., HWFR-D (0.3) has a larger number of search blocks than HWFR-D (0.2) and HWFR-D (0.1). In contrast, when the fixed convenient visit threshold in HWFR-S is set lower, there is a slight increase in the number of search blocks for HWFR-S. Hence, HWFR-S (0.6) achieves the lowest number of search blocks in Figure 20a–c. As mentioned above, if a larger error and probability variation on the AP value is given in the simulation, the difference between the calculated the first-/second-level weight value (based on the AP values of the four target features of each recognized person in every block) and its correct value becomes larger. Hence, the HWF algorithm and its variants have a higher probability of deploying the UAV to an incorrect block to search for the search target, which also results in a larger number of search blocks, as shown in Figure 20a–c.

4.2.3. Average Search Time

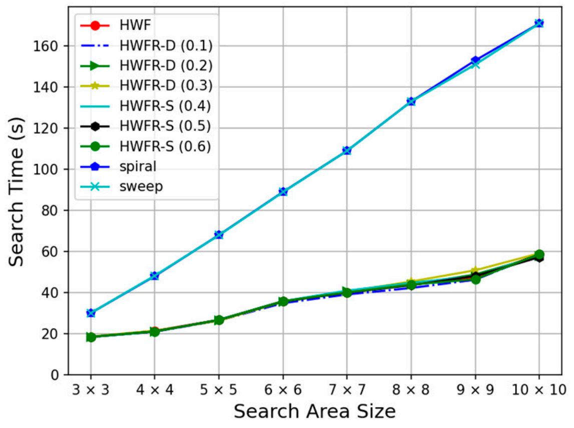
The search time in this study is defined as the sum of the UAV flying time spent on the search path and the UAV hovering and image capture time for all visited blocks. In Figures 19 and 20, the spiral and sweep algorithms exhibit nearly identical and linear growth in both the search path length and the number of search blocks. As a result, the calculated search times for these two algorithms are nearly identical, as shown in Figure 21a–c.



(**a**)



(**b**)



(**c**)

Figure 21. Average search time vs. search area size for (**a**) 10%, 5%; (**b**) 20%, 10%; (**c**) 30%, 15% error and probability variations at altitudes h_0 and h_1.

The HWF, HWFR-S, and HWFR-D algorithms exhibit a significant difference in their search times compared to the spiral and sweep algorithms. Under the parameters given in Table 3, with a block size of 10 × 10, their search time is effectively reduced by over 80%, 75%, and 60%, as shown in Figure 21a–c, respectively. In situations where the convenient visit threshold for HWFR-S and the weight difference for HWFR-D are not set appropriately, the search time for the HWF algorithm can be lower than in some HWFR-S and HWFR-D algorithms.

The HWFR-S algorithm exhibits its longest search time among the three fixed convenient visit thresholds when the threshold is set to the highest value of 0.6. The search time for HWFR-S (0.6) is also higher than that of the original HWF algorithm. When the fixed convenient visit threshold of HWFR-S is lowered to 0.5, the average search time is not significantly different from that when the threshold is set to 0.4. The HWFR-D algorithm exhibits its longest search time when the weight difference is set to the highest value of 0.3, compared to both the HWF algorithm and HWFR-S with three different fixed convenient visit thresholds. However, when the HWFR-D weight difference is set to 0.2, it results in a shorter search time among the HWF algorithm and HWFR-S with three different fixed convenient visit thresholds. Furthermore, as the weight difference of HWFR-D decreases, the search time becomes shorter. When the weight difference is set to 0.1, the HWFR-D algorithm, as shown in Figure 21a–c, achieves the shortest search time among all considered algorithms. As mentioned above, if a larger error and probability variation on the AP value is given in the simulation, the HWF algorithm and its variants have a higher probability of deploying the UAV to an incorrect block to search for the search target, which also results in a longer search time, as shown in Figure 21a–c.

4.2.4. Average Search Accuracy

When the search algorithm encounters a block having a recognized person with a weight greater than the target threshold, it concludes that the search target has been found. The search accuracy in this study is defined as the probability that the target found by the search algorithm is the same as the actual search target given in the testing data.

As shown in Figure 22a–c, the spiral and sweep algorithms do not prioritize the searching order of blocks that may potentially contain the target. With (10%, 5%), (20%, 10%), and (30%, 15%) error and probability variations on features in the testing data at altitudes h_0 and h_1, respectively, it is observed that these two search algorithms, when applied to the 10 × 10 area size, achieve 50%, 40%, and 30% accuracy, respectively, in determining the actual search target after averaging over 10,000 testing data. The reason for this low accuracy is that these two algorithms often find a false target due to the high percentages of errors and probability variations when they encounter blocks with weights above the target threshold. Moreover, the HWF algorithm and its variants also suffer from lower search accuracy if a larger error and probability variation on the AP value is given in the simulation, which is shown in Figure 22a–c.

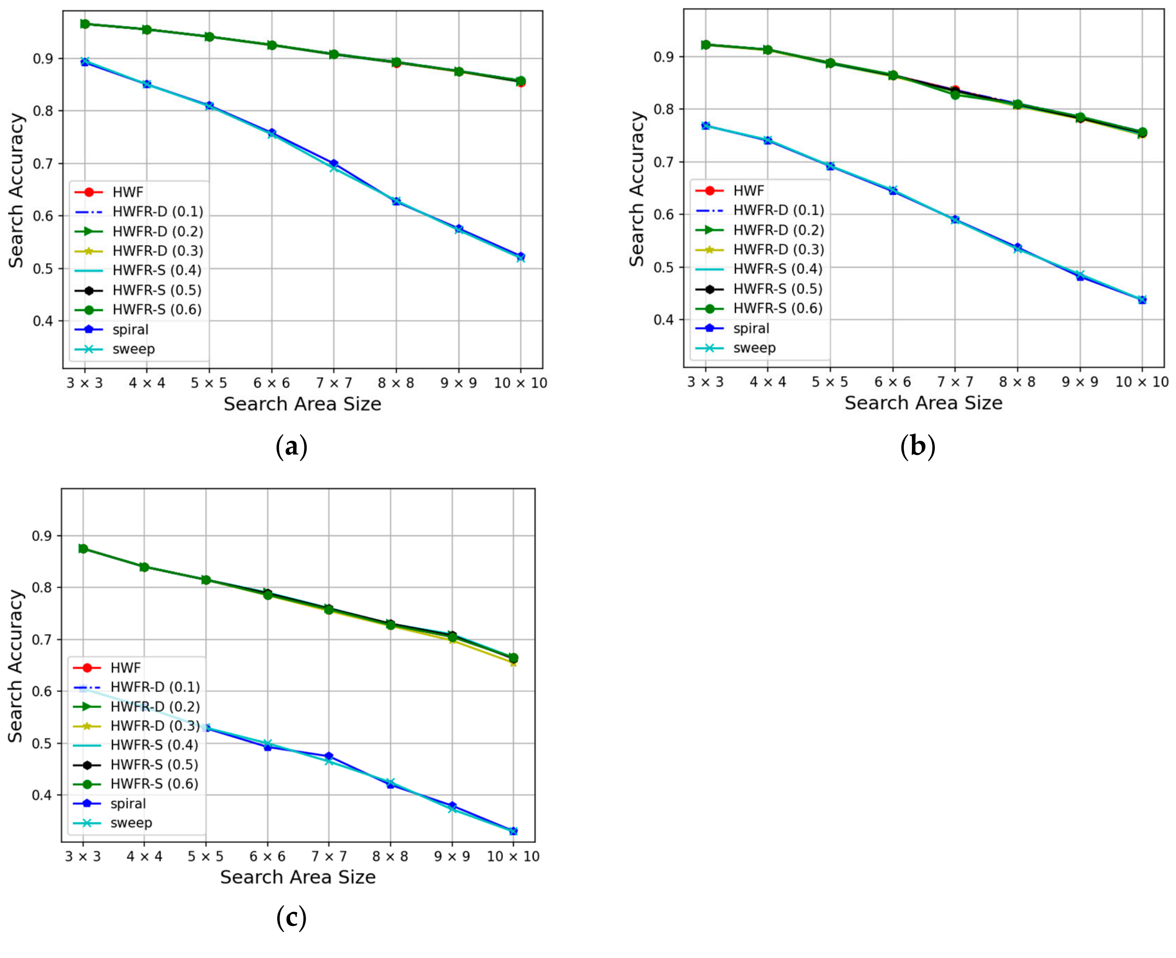


Figure 22. Average search accuracy vs. search area size for (**a**) 10%, 5%; (**b**) 20%, 10%; (**c**) 30%, 15% error and probability variations at altitudes h_0 and h_1.

HWF, HWFR-S, and HWFR-D all share the characteristic of prioritizing the search order of blocks that is most likely to contain the target. Therefore, compared to search algorithms such as spiral and sweep, these algorithms achieve significantly higher accuracy.

Even with an area size of 10 × 10, the HWF-based algorithms maintain accuracy higher than 85%, 75%, and 65%, respectively, as shown in Figure 22a–c. Furthermore, as observed from Figure 20, different values of the convenient visit threshold and weight difference do not significantly affect the average accuracy of HWFR-S and HWFR-D after averaging over 10,000 testing data. Consequently, these HWFR-S and HWFR-D algorithms can reduce the search path length and decrease the search time, which in turn enhances the search efficiency.

In summary, the higher the error and probability variation applied to the AP values and each feature of the person is, the longer the search path length, the larger the number of search blocks, the longer the search time, and the lower the search accuracy of the HWF algorithm and all its HWFR-S and HWFR-D variants.

5. System Implementation

5.1. Software and Hardware

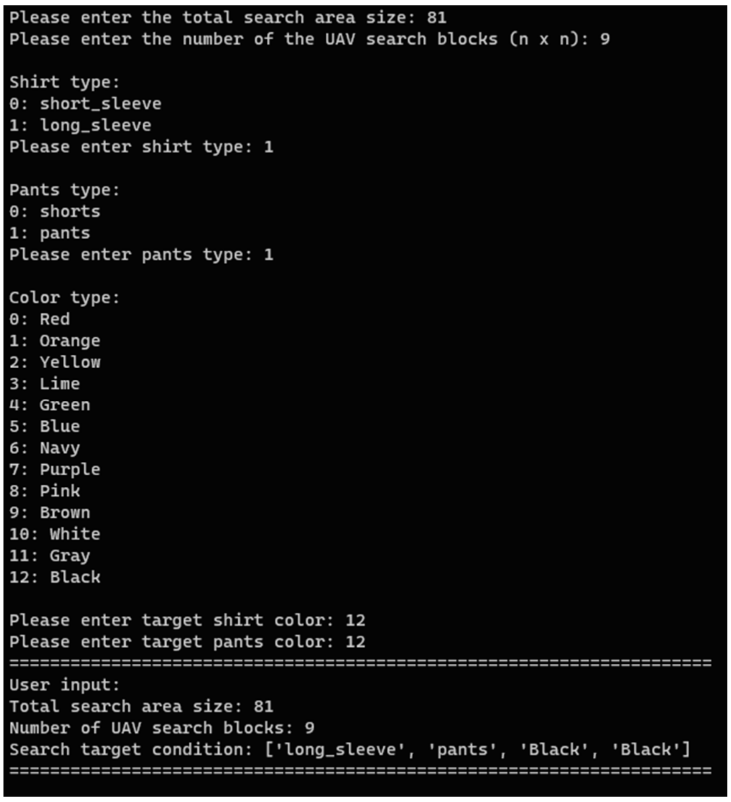
The searching UAV in this study adopts Pixhawk2.4.8 [39] as the core platform, enhanced by the M8N high-precision GNSS/GPS positioning and navigation module, a GY-271M compass sensor module, a GY-US42V2 ultrasonic sensor range finder, 2212/13T brushless motors, a DEVO-RX1002 radio receiver, a DEVO10 remote controller, a PPM remote decoder, an electronic speed controller, a voltage indicator, an 8-inch nylon propeller, and a 915 MHz 100 mW servo-side transmitter. The communication protocol employed is 2.4 GHz (DSSS). Additionally, an Nvidia Jetson Nano [34] and USB camera are mounted on the UAV. The server environment operates on Windows 10, with hardware specifications including an Intel® Core™ i7-8700 CPU, 16 GB of RAM, and a GeForce RTX 2080 TI GPU. In terms of software, Python serves as the primary programming language, while DroneKit-Python [40] is employed for UAV flight control, and the UAV platform is Mission Planner [41,42].

5.2. Screenshots of the Implemented System

As shown in Figure 23, the user inputs the values of the total search area size, the number of search blocks, and the features of the search target, including the types and colors of the clothing and pants, at the server side. The search target to be found is wearing a black long-sleeved shirt and black pants. After the user inputs the search information, the server computes the UAV flight altitudes h_0 and h_1 and transmits them to the Jetson Nano on the searching UAV to dispatch the UAV to fly to the center of search area at altitude h_0. The Jetson Nano on the UAV captures images at altitude h_0 and transmits them back to the server. The server first partitions the search area into 9 blocks, executes the weighting subroutine to calculate the first-level weights of all blocks, and calculates the HWF block traversal order for the UAV, which is block [35.06] $\rightarrow$ block [9.0] $\rightarrow$ block [3.39] in this example, as depicted in Figure 24. Subsequently, the server transmits this block traversal order to the Jetson Nano.

After receiving the block traversal order, the Jetson Nano commands the UAV to fly and descend to the center of the block with the highest weight, i.e., block [35.06], at altitude h_1. The UAV further captures images of this block and transmits them back to the server. Using the weighting subroutine, the server calculates the weight value of each recognized person in this block and determines whether the second-level weight value is larger than the search target threshold. In Figure 25, since the person is wearing a red long-sleeved shirt and black pants, the computed second-level weight of this person is 0.7534, which is lower than the search target threshold of 0.8. Hence, the UAV continues the HWF search in the next block with a first-level weight of 9.0.

In Figure 26, because the recognized person in block [9.0] is wearing a black long-sleeved shirt and black pants, the calculated second-level weight value for this person is 0.9052, which is greater than the target threshold of 0.8. Hence, the algorithm determines that the search target has been found. The Jetson Nano on the UAV then transmits photos of the person found in this block, along with the location coordinates, back to the server to notify the user, which concludes this search mission.



```
Please enter the total search area size: 81
Please enter the number of the UAV search blocks (n x n): 9

Shirt type:
0: short_sleeve
1: long_sleeve
Please enter shirt type: 1

Pants type:
0: shorts
1: pants
Please enter pants type: 1

Color type:
0: Red
1: Orange
2: Yellow
3: Lime
4: Green
5: Blue
6: Navy
7: Purple
8: Pink
9: Brown
10: White
11: Gray
12: Black

Please enter target shirt color: 12
Please enter target pants color: 12
=====================================================================
User input:
Total search area size: 81
Number of UAV search blocks: 9
Search target condition: ['long_sleeve', 'pants', 'Black', 'Black']
=====================================================================
```

Figure 23. User inputs search information at server.

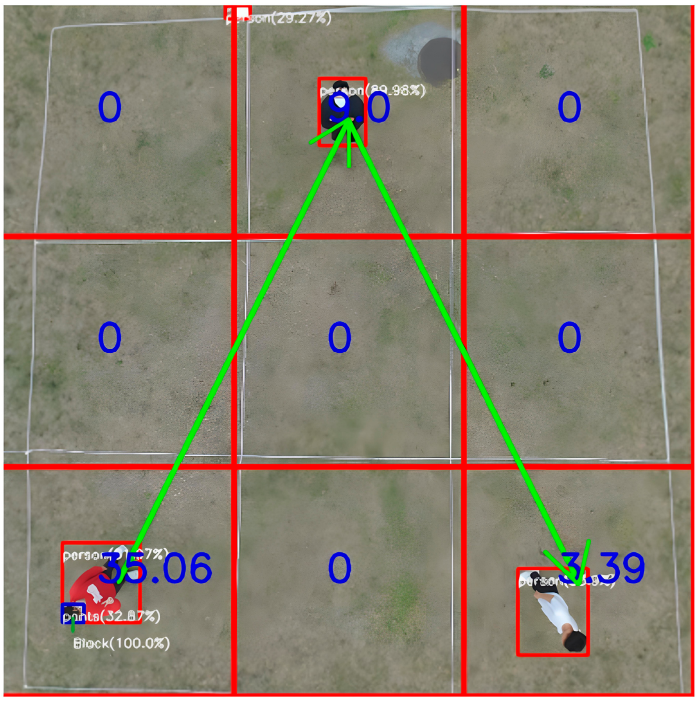


Figure 24. The HWF block traversal order (green line) in search area. The red lines and the blue values indicate the boundaries and the first-level weights of all blocks respectively.

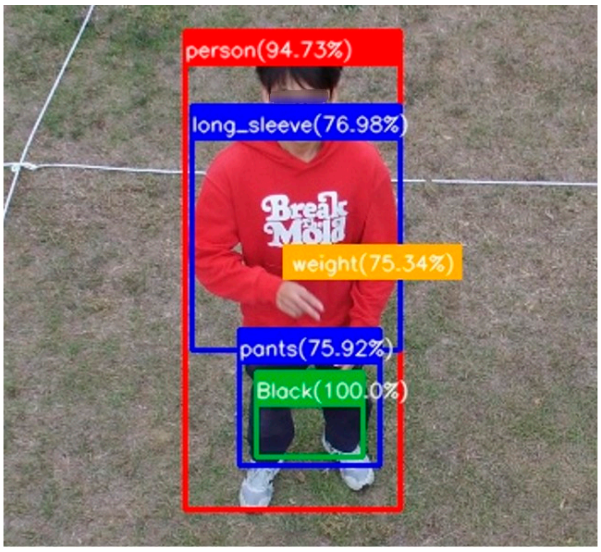


Figure 25. Recognized results for the person in the block with weight 35.06 at altitude h_1.

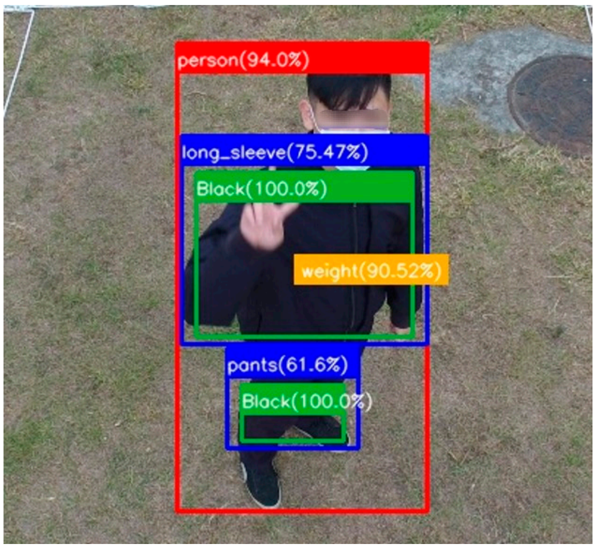


Figure 26. Recognized results for the person in the block with weight 9.0 at altitude h_1.

5.3. Limitations of the Proposed System

There are important prerequisites for the integrated YOLOv5 and hierarchical HWF path planning framework proposed in this study, which are as follows.

1. The ideal search area must approximate a rectangle. If it is a concave or convex polygon or any irregular shape, the input range for HWF must be the smallest bounding rectangle that encompasses the entire search area. This would expand the search area, potentially including many non-search areas, leading to longer search paths and times, as well as increased power consumption for the UAV.
2. The ideal altitude for the search area should be consistent across a single horizontal plane. This ensures that when the UAV captures the image of the entire search area at altitude h_0, the distances between the UAV and the center points of different blocks are similar. Hence, the relationship among the block weight values obtained from the human body and clothing recognition at the first level will closely approximate those of the real search target. Further, using HWF path planning to visit blocks at the second level with the highest block weight value first and subsequently recognizing the results at the second level will yield more accurate outcomes. Conversely, if the altitudes of

the center points of different blocks are not on the same horizontal plane, the UAV will be closer to blocks at higher altitudes. This results in clearer, more magnified images of human bodies, leading to better recognition results. Consequently, a block with higher block weight values might be prioritized in the HWF path planning algorithm. If the search target is not within this block, it could result in longer search paths and times.

3. Since the UAV captures images at altitude h_0 in the first level, it must cover the entire area. Due to the limited resolution of the camera, the search area cannot be too expansive. Otherwise, the captured images of human bodies would appear smaller and blurrier, leading to poorer recognition results and subsequently affecting the accuracy of HWF path planning.

5.4. Performance Comparison of YOLOv5 and YOLOv8

YOLOv5 was released in June 2020. The YOLOv5 model has Darknet 53 as its backbone and its design focuses on enhancing the detection of objects at different scales, which improves the performance on objects of varying sizes. YOLOv8 is the latest version of the YOLO family, developed by Ultralytics, who also created the YOLOv5 model. It introduces numerous architectural changes over YOLOv5. Unlike YOLOv5, YOLOv8 is an anchor-free model, meaning that it directly predicts the center of an object instead of the offset from a known anchor box. The study conducted by [43] involved a comparative analysis of the performance of YOLOv5 and YOLOv8 in aerial human detection using unmanned aerial vehicles. The research utilized a pedestrian dataset obtained from Roboflow, consisting of 828 aerial images for model training and 233 images for validation. The experimental results revealed that the YOLOv8 model exhibited higher precision and F1-scores compared to the YOLOv5 model, with differences of 2.82% and 0.98%, respectively. However, in terms of recall performance, YOLOv5 surpasses the YOLOv8 model by 0.54%. In [44], a comparison was made between various versions of YOLO, specifically YOLOv5 to YOLOv8, based on their mean average precision (mAP) scores. The study involved the training of models using 2415 images, and 303 images were allocated for both validation and testing purposes. The images were categorized into five classes. The research findings indicated that YOLOv5 achieved the highest average detection accuracy in terms of the mAP metric, whereas YOLOv8 performed the poorest among the four versions. Considering these results collectively, it is evident that the choice of dataset significantly influences the performance of different YOLO versions. Therefore, for future work, we are contemplating the utilization of the updated YOLOv8 to train our human detection model.

6. Conclusions

In this study, to reduce the search time and increase the search accuracy of search and rescue operations, an integrated YOLOv5 and HWF framework has been proposed. It combines the YOLOv5 model to automatically recognize the search target in real time and the hierarchical HWF path planning algorithm to dispatch a UAV to capture images of the search target at different altitudes. Two improved search algorithms, HWFR-S and HWFR-D, which incorporate the concepts of the convenient visit threshold and the weight difference, respectively, have been further proposed to resolve the issue of the lengthy and redundant flight paths of HWF. YOLOv5 has been trained by the VisDrone2019 dataset and the drone-clothing dataset for human body and clothing/pant recognition, respectively. The results show that the drone-clothing model converges faster and achieves higher classification metric values than the VisDrone2019 model does. According to the simulation results, the HWF, HWFR-S, and HWFR-D search algorithms proposed in this study not only effectively reduce the length of the UAV's search path and the number of search blocks but also significantly decrease the search time required for the UAV to locate the search target, with a much higher search accuracy than the two traditional search algorithms. Moreover, this integrated YOLOv5 and HWF framework has been implemented and tested in a real scenario; it has been shown to reduce the UAV's power consumption and enhance the efficiency of search and rescue operations.

In the future, we will address the issues and limitations mentioned in Section 5.3, for example, the irregular shapes of search and rescue areas, the different altitudes of areas, and areas that exceed the resolution range of the UAV camera. Additionally, to reduce search and rescue times, the possibility of utilizing multiple UAVs in collaborative search and rescue operations will be considered. Further, we will evaluate the utilization of the updated YOLOv8 to train our human detection model and compare its performance with that of YOLOv5 as another future task. By refining the integrated YOLOv5 and hierarchical HWF path planning system proposed in this study, we aim to develop a more versatile UAV search and rescue system in the future.

Author Contributions: Conceptualization, I.-C.C.; methodology, I.-C.C., H.-F.C., Y.-W.C., M.-T.H., W.-F.W. and D.-Y.Y.; software, H.-F.C., Y.-W.C., M.-T.H., W.-F.W. and D.-Y.Y.; validation, I.-C.C. and C.-E.Y.; formal analysis, I.-C.C.; resources, C.-E.Y.; data curation, H.-F.C., Y.-W.C., M.-T.H., W.-F.W. and D.-Y.Y.; writing—original draft preparation, Y.-H.H., I.-C.C. and C.-E.Y.; writing—review and editing, Y.-H.H., I.-C.C. and C.-E.Y.; supervision, I.-C.C. and C.-E.Y.; project administration, I.-C.C. and C.-E.Y.; funding acquisition, I.-C.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Science and Technology Council, Taiwan, grant number NSTC 112-2221-E-018-009. The APC was funded by NSTC 112-2221-E-018-009.

Data Availability Statement: Data are contained within the article.

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

References

1. Sahingoz, O.K. Networking models in flying ad-hoc networks (FANETs): Concepts and Challenges. *J. Intell. Robot. Syst.* **2014**, *74*, 513–527. [CrossRef]
2. Menouar, H.; Guvenc, I.; Akkaya, K.; Uluagac, A.S.; Kadri, A.; Tuncer, A. UAV-enabled intelligent transportation systems for the smart city: Applications and challenges. *IEEE Commun. Mag.* **2017**, *55*, 22–28. [CrossRef]
3. Aasen, H. UAV spectroscopy: Current sensors, processing techniques and theoretical concepts for data interpretation. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 22–27 July 2018; pp. 8809–8812.
4. Ezequiel, C.A.F.; Cua, M.; Libatique, N.C.; Tangonan, G.L.; Alampay, R.; Labuguen, R.T.; Favila, C.M.; Honrado, J.L.E.; Canos, V.; Devaney, C.; et al. UAV aerial imaging applications for post-disaster assessment, environmental management and infrastructure development. In Proceedings of the International Conference on Unmanned Aircraft Systems, Orlando, FL, USA, 27–30 May 2017; pp. 274–283.
5. Zhang, Y.; Li, S.; Wang, S.; Wang, X.; Duan, H. Distributed bearing-based formation maneuver control of fixed-wing UAVs by finite-time orientation estimation. *Aerosp. Sci. Technol.* **2023**, *136*, 108241. [CrossRef]
6. Zheng, Q.; Zhao, P.; Li, Y.; Wang, H.; Yang, Y. Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification. *Neural Comput. Applic.* **2021**, *33*, 7723–7745. [CrossRef]
7. Mao, Y.; Sun, R.; Wang, J.; Cheng, Q.; Kiong, L.C.; Ochieng, W.Y. New time-differenced carrier phase approach to GNSS/INS integration. *GPS Solut.* **2022**, *26*, 122. [CrossRef]
8. Zhang, X.; Pan, W.; Scattolini, R.; Yu, S.; Xu, X. Robust tube-based model predictive control with Koopman operators. *Automatica* **2022**, *137*, 110114. [CrossRef]
9. Narayanan, S.S.K.S.; Tellez-Castro, D.; Sutavani, S.; Vaidya, U. SE(3) (Koopman-MPC: Data-driven learning and control of quadrotor UAVs. *IFAC-PapersOnLine* **2023**, *56*, 607–612. [CrossRef]
10. Cao, B.; Zhang, W.; Wang, X.; Zhao, J.; Gu, Y.; Zhang, Y. A memetic algorithm based on two_Arch2 for multi-depot heterogeneous-vehicle capacitated arc routing problem. *Swarm Evol. Comput.* **2021**, *63*, 100864. [CrossRef]
11. Erdelj, M.; Natalizio, E. UAV-assisted disaster management: Applications and open issues. In Proceedings of the International Conference on Computing, Networking and Communications, Kauai, HI, USA, 15–18 February 2016; pp. 1–5.
12. Mukherjee, A.; De, D.; Dey, N.; Crespo, R.G.; Herrera-Viedma, E. DisastDrone: A Disaster Aware Consumer Internet of Drone Things System in Ultra-Low Latent 6G Network. *IEEE Trans. Consum. Electron.* **2023**, *69*, 38–48. [CrossRef]
13. Pasandideh, F.; da Costa, J.P.J.; Kunst, R.; Islam, N.; Hardjawana, W.; Pignaton de Freitas, E. A Review of Flying Ad Hoc Networks: Key Characteristics, Applications, and Wireless Technologies. *Remote Sens.* **2022**, *14*, 4459. [CrossRef]
14. Majeed, A.; Hwang, S.O. A Multi-Objective Coverage Path Planning Algorithm for UAVs to Cover Spatially Distributed Regions in Urban Environments. *Aerospace* **2021**, *8*, 343. [CrossRef]

15. Das, L.B.; Das, L.B.; Lijiya, A.; Jagadanand, G.; Aadith, A.; Gautham, S.; Mohan, V.; Reuben, S.; George, G. Human Target Search and Detection using Autonomous UAV and Deep Learning. In Proceedings of the IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), Bali, Indonesia, 7–8 July 2020; pp. 55–61. [CrossRef]
16. Bandeira, T.W.; Coutinho, W.P.; Brito, A.V.; Subramanian, A. Analysis of Path Planning Algorithms Based on Travelling Salesman Problem Embedded in UAVs. In Proceedings of the Brazilian Symposium on Computing Systems Engineering (SBESC), Fortaleza, Porto Alegre, Brazil, 3–6 November 2015; pp. 70–75. [CrossRef]
17. Jain, A.; Ramaprasad, R.; Narang, P.; Mandal, M.; Chamola, V.; Yu, F.R.; Guizan, M. AI-Enabled Object Detection in UAVs: Challenges, Design Choices, and Research Directions. *IEEE Netw.* **2021**, *35*, 129–135. [CrossRef]
18. Yu, X.; Jin, S.; Shi, D.; Li, L.; Kang, Y.; Zou, J. Balanced Multi-Region Coverage Path Planning for Unmanned Aerial Vehicles. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON, Canada, 11–14 October 2020; pp. 3499–3506.
19. Yaguchi, Y.; Tomeba, T. Region Coverage Flight Path Planning Using Multiple UAVs to Monitor the Huge Areas. In Proceedings of the IEEE International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 15–18 June 2021; pp. 1677–1682.
20. Kurdi, H.A.; Aloboud, E.; Alalwan, M.; Alhassan, S.; Alotaibi, E.; Bautista, G.; How, J.P. Autonomous Task Allocation for Multi-UAV Systems Based on the Locust Elastic Behavior. *Appl. Soft Comput.* **2018**, *71*, 110–126. [CrossRef]
21. Alotaibi, E.T.; Alqefari, S.S.; Koubaa, A. LSAR-Multi-UAV Collaboration for Search and Rescue Missions. *IEEE Access* **2019**, *7*, 55817–55832. [CrossRef]
22. Cabreira, T.; Brisolara, L.; Ferreira, P.R., Jr. Survey on Coverage Path Planning with Unmanned Aerial Vehicles. *Drones* **2019**, *3*, 4. [CrossRef]
23. Jünger, M.; Reinelt, G.; Rinaldi, G. The Traveling Salesman Problem. In *Handbooks in Operations Research and Management Science*; Elsevier B.V.: Amsterdam, The Netherlands, 1995; Volume 7, pp. 225–330.
24. Ali, M.; Md Rashid, N.K.A.; Mustafah, Y.M. Performance Comparison between RGB and HSV Color Segmentations for Road Signs Detection. *Appl. Mech. Mater.* **2013**, *393*, 550–555. [CrossRef]
25. Haritha, D.; Bhagavathi, C. Distance Measures in RGB and HSV Color Spaces. In Proceedings of the 20th International Conference on Computers and Their Applications (CATA 2005), New Orleans, LA, USA, 16–18 March 2005.
26. Pooja, K.S.; Shreya, R.N.; Lakshmi, M.S.; Yashika, B.C.; Rekha, B.N. Color Recognition using K-Nearest Neighbors Machine Learning Classification Algorithm Trained with Color Histogram Features. *Int. Res. J. Eng. Technol. (IRJET)* **2021**, *8*, 1935–1936.
27. Pradeep, A.G.; Gnanapriya, M. Novel Contrast Enhancement Algorithm Using HSV Color Space. *Int. J. Innov. Technol. Res.* **2016**, *4*, 5073–5074.
28. Krishna, S.L.; Chaitanya, G.S.R.; Reddy, A.S.H.; Naidu, A.M.; Poorna, S.S.; Anuraj, K. Autonomous Human Detection System Mounted on a Drone. In Proceedings of the 2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET), Chennai, India, 21–23 March 2019; pp. 335–338. [CrossRef]
29. Mliki, H.; Bouhlel, F.; Hammami, H. Human activity recognition from UAV-captured video sequences. *Pattern Recognit.* **2020**, *100*, 107140. [CrossRef]
30. Safadinho, D.; Ramos, J.; Ribeiro, R.; Filipe, V.; Barroso, J.; Pereira, A. UAV Landing Using Computer Vision Techniques for Human Detection. *Sensors* **2020**, *20*, 613. [CrossRef] [PubMed]
31. Lygouras, E.; Santavas, N.; Taitzoglou, A.; Tarchanidis, K.; Mitropoulos, A.; Gasteratos, A. Unsupervised Human Detection with an Embedded Vision System on a Fully Autonomous UAV for Search and Rescue Operations. *Sensors* **2019**, *19*, 3542. [CrossRef]
32. Do, M.-T.; Ha, M.-H.; Nguyen, D.-C.; Thai, K.; Ba, Q.-H.D. Human Detection Based Yolo Backbones-Transformer in UAVs. In Proceedings of the International Conference on System Science and Engineering (ICSSE), Ho Chi Minh, Vietnam, 27–28 July 2023; pp. 576–580. [CrossRef]
33. Wijesundara, D.; Gunawardena, L.; Premachandra, C. Human Recognition from High-altitude UAV Camera Images by AI based Body Region Detection. In Proceedings of the Joint 12th International Conference on Soft Computing and Intelligent Systems and 23rd International Symposium on Advanced Intelligent Systems (SCIS & ISIS), Ise, Japan, 29 November—2 December 2022; pp. 1–4. [CrossRef]
34. Jetson Nano Developer Kit | NVIDIA. Available online: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano-developer-kit/ (accessed on 1 December 2023).
35. Itkin, M.; Kim, M.; Park, Y. Development of Cloud-Based UAV Monitoring and Management System. *Sensors* **2016**, *16*, 1913. [CrossRef]
36. Geng, X.; Chen, Z.; Yang, W.; Shi, D.; Zhao, K. Solving the Traveling Salesman Problem Based on an Adaptive Simulated Annealing Algorithm with Greedy Search. *Appl. Soft Comput.* **2011**, *11*, 3680–3689. [CrossRef]
37. OpenCV—Open Computer Vision Library. Available online: https://opencv.org/ (accessed on 1 December 2023).
38. VisDrone-Dataset-github. Available online: https://github.com/VisDrone/VisDrone-Dataset (accessed on 1 December 2023).
39. Pixhawk. Available online: https://pixhawk.org/ (accessed on 1 December 2023).
40. Welcome to DroneKit-Python's Documentation. Available online: https://dronekit-python.readthedocs.io/en/latest/ (accessed on 1 December 2023).
41. Mission Planner Home—Mission Planner Documentation (ardupilot.org). Available online: https://ardupilot.org/planner/ (accessed on 1 December 2023).

42. Suparnunt, C.; Boonvongsobhon, C.; Eounes Baig, F.; Leelalerthpat, P.; Hematulin, W.; Jarawan, T.; Kamsing, P. Practical Parallel of Autonomous Unmanned Aerial Vehicle by Mission Planner. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Kuala Lumpur, Malaysia, 17–22 July 2022; pp. 7831–7834. [CrossRef]
43. Sary, I.P.; Andromeda, S.; Armin, E.U. Performance Comparison of YOLOv5 and YOLOv8 Architectures in Human Detection using Aerial Images. *Ultim. Comput. J. Sist. Komputer.* **2023**, *15*, 8–13. [CrossRef]
44. Gašparović, B.; Mauša, G.; Rukavina, J.; Lerga, J. Evaluating YOLOV5, YOLOV6, YOLOV7, and YOLOV8 in Underwater Environment: Is There Real Improvement? In Proceedings of the 8th International Conference on Smart and Sustainable Technologies (SpliTech), Split/Bol, Croatia, 20–23 June 2023; pp. 1–4. [CrossRef]