Hindawi
Advances in Multimedia
Volume 2022, Article ID 8908122, 11 pages
https://doi.org/10.1155/2022/8908122

Research Article

UAV Intelligent Control Based on Machine Vision and Multiagent Decision-Making

Zishan Huang

School of Electrical and Electronic Engineering, Hubei University of Technology, Wuhan 430068, China

Correspondence should be addressed to Zishan Huang; 2010211231@hbut.edu.cn

Received 8 March 2022; Revised 11 April 2022; Accepted 29 April 2022; Published 27 May 2022

Academic Editor: Qiangyi Li

In order to improve the effect of UAV intelligent control, this paper will improve machine vision technology. Moreover, this paper adds scale information on the basis of the LSD algorithm, uses the multiline segment standard to merge these candidate line segments for intelligent recognition, and uses the LSD detection algorithm to improve the operating efficiency of the UAV control system and reduce the computational complexity. In addition, this paper combines machine vision technology and multiagent decision-making technology for UAV intelligent control and builds an intelligent control system, which uses intelligent machine vision technology for recognition and multiagent decision-making technology for motion control. The research results show that the UAV intelligent control system based on machine vision and multiagent decision-making proposed in this paper can achieve reliable control of UAVs and improve the work efficiency of UAVs.

1. Introduction

Unmanned aerial vehicles can complete flight missions through wireless remote control or even autonomous control. Compared with ordinary manned aircraft, UAV has many advantages such as simple structure, flexible operation, low cost, easy manufacturing, and easy maintenance [1]. At the same time, the UAV can be remotely controlled by wireless equipment and will not endanger the life and safety of the operator in an accident [2]. Therefore, UAVs are widely used in various fields such as civil and military. In civil use, UAVs can be used for air transportation, remote aerial photography, traffic patrol, water conservancy monitoring, and forest fire fighting [3]. In the military, UAVs can be used for enemy reconnaissance, electronic interference, target positioning, and precise strikes on specific targets. According to the different body structures, UAV can be divided into two categories: fixed wing and rotary wing. Fixed-wing UAVs mainly include two types: propeller type and jet type. The principle is to use the thrust or pulling force generated by the engine to make the aircraft fly horizontally while using the lift generated by the wings to maintain the vertical motion of the body. Rotor UAVs are divided into two types: single-rotor and multirotor, and common multirotor UAVs have four-rotor, six-rotor, and eight-rotor forms. Single-rotor UAVs generally need a separate tail to balance the torque generated by the main wing, while multirotor UAVs can cancel each other's rotation torque due to the opposite rotation of adjacent wings [4]. Therefore, the structure of the multirotor UAVs will be simpler, and the maneuverability will be more superior.

This paper combines machine vision technology and multiagent decision-making technology to study the intelligent control of UAVs, uses intelligent machine vision technology for recognition, and uses multiagent decision-making technology for motion control to improve the motion effect of UAVs.

2. Related Work

Intelligent control belongs to the advanced stage of the development of control theory. The use of intelligent control methods can solve the control problems of some complex systems that cannot be handled by traditional control methods. Different from the traditional control method, which relies heavily on the precise mathematical model of

the controlled object [5], the intelligent control method can be applied to the control of uncertain objects with unknown models or model parameters and structural changes. At the same time, the intelligent control method also has good advantages for the control of systems with strong nonlinearity and complex tasks. With the continuous improvement and development of intelligent control theory, intelligent control has been successfully applied in many engineering fields and has become one of the most attractive and valuable technologies in the field of control technology. Rotor UAVs have complex structures and strong coupling between different axes, making it difficult to obtain accurate mathematical models. This is where intelligent control methods are good. The application of intelligent control methods to the attitude control of rotary wing drones can make up for the shortcomings of traditional control methods and improve control performance. In recent years, more and more scholars have begun to pay attention to the application of intelligent control methods in the attitude control of rotor drones, trying to improve the effect of rotor drone attitude control and truly realize intelligent control. Commonly used intelligent control methods include fuzzy control, neural network control, genetic algorithm, and ant colony algorithm. Fuzzy control is based on fuzzy set theory, fuzzy linguistic variables, and fuzzy logic reasoning and simulates human approximate reasoning and decision-making process [6]. The core part of the fuzzy control method is the determination of fuzzy rules. Generally speaking, fuzzy rules can be determined based on expert experience or experiments [7]. Literature [8] took the three-degree-of-freedom helicopter system as the research object, respectively, designed PID controller, LQR controller, and fuzzy controller to control the helicopter's attitude, and compared the control effects of the three controllers through simulation and verified the fuzzy. The advantages of control: Literature [9] designed an intelligent four-rotor control system based on fuzzy logic. Literature [10] designed four fuzzy controllers for altitude, pitch angle, yaw angle, and tilt angle. The structures of these fuzzy controllers are all relatively simple, the fuzzy rules are determined by expert experience, and then the outputs of the four fuzzy controllers are used as the reference values of the driving voltages of the four motors to control the attitude of the quadrotor. Finally, the effectiveness of the control method is verified by simulation. Literature [11] takes into account the influence of air resistance and rotational torque on the quadrotor, establishes a dynamic model, and then uses a fuzzy control method to adjust the parameters of the PID controller. The design of the fuzzy controller is to find the input deviation, the deviation change rate and for the relationship between the three parameters of PID, the fuzzy controller designed a total of 49 fuzzy rules, and then the simulation verified that the control method has a better control effect. The neural network is a way of simulating human thinking. Although the structure of a single neuron is relatively simple and its functions are limited, the behavior that can be achieved by a network system composed of a large number of neurons is extremely colorful [12]. With the deepening of neural network control research, this method has become an important branch of intelligent control, and it has a wide range of applications in solving complex nonlinear, time-varying, and uncertain system control problems. Compared with traditional control methods, the research of neural network algorithms in the attitude control of rotary wing UAV is in its infancy [13]. In the control of rotary wing UAV, the neural network is often used to identify some unknown parameters to supplement and optimize traditional control methods such as PID, LQR, etc. Literature [14] designed a neural network. The PID control system has designed three neural networks for pitch angle, yaw angle, and roll angle. PID controller: the input of each neural network is the error of the corresponding attitude angle and the rate of change of the error, and the output is the correction value of the three parameters of the PID controller. The entire network adopts a four-layer neural network structure. Unfortunately, they did not give the training process of the neural network but only the network parameters after the training. Finally, the simulation demonstrated the superiority of the design method and other traditional methods in the control performance and carried out the method on the real object. The experiment verified the feasibility of the method. Literature [15] uses a neural network to modify PID parameters and gives the training process of a neural network based on ideal experimental data. Literature [16] designed a quadrotor control method based on neural network output feedback for the complex situation of the quadrotor in an outdoor environment. This method first designed a multilayer neural network to learn the dynamic characteristics of UAV online, and then a neural network is designed to provide feedback on the position and attitude of the UAV as well as external interference, and finally, the feedback information is sent to the feedback controller for control. Literature [17] verifies the convergence of the main parameters of the system and analyzes it through simulation experiments. The control performance of the strategy: Literature [18] proposed a robust adaptive controller based on radial neural network interference compensation for the symmetrical structure of the six-rotor attitude control problem, and the simulation verified the method's suppression effect on interference. Literature [19] proposed a PIDNN control method combining neural network ideas and PID principles. Since then, many scholars have applied the method to the attitude control of three-degree-of-freedom helicopters and quadrotors. Simulations have verified that the method is relative to the effectiveness of PID control methods.

3. Intelligent Machine Vision Optical Inspection Algorithm

LSD (Line Segment Detector) is a linear timeline segment detector that can provide subpixel accuracy results. It can process any digital image without any parameter adjustment, and at the same time, it can control the number of its own error detection: on average, each image allows one error alarm. Compared with the classic Hough transform, the LSD line segment detection algorithm not only improves the accuracy but also greatly reduces the computational complexity and greatly improves the speed.

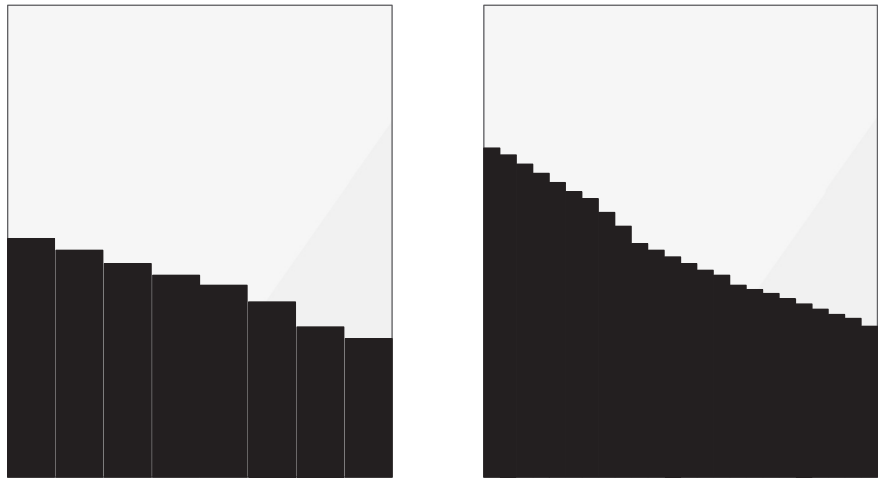

Figure 1: The sawtooth effect at different scales.

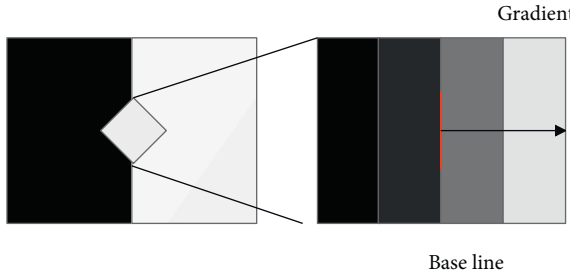

Figure 2: Gradient and gradient angle.

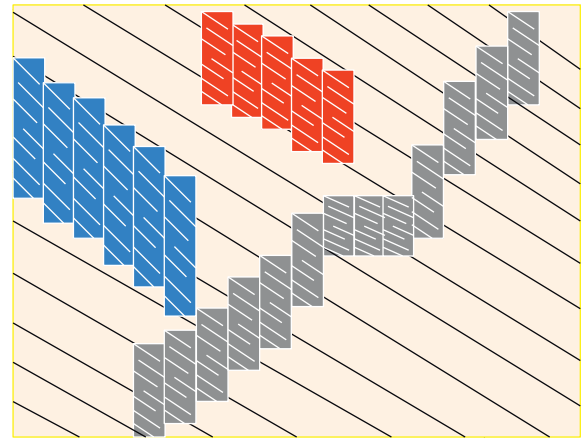

Figure 3: Line segment support area.

The flow of the entire algorithm is roughly as follows:

(1) The algorithm reduces the image to 80% of the original through Gaussian downsampling (both the length and width are reduced to 80% of the original, and the total pixels become 64% of the original). The purpose of this is to reduce or eliminate the aliasing effect that often appears in the image, as shown in Figure 1:

(2) The algorithm calculates the gradient amplitude and gradient angle of each pixel in the image, as shown in Figure 2. The algorithm uses a 2×2 template to calculate the gradient and gradient angle. The smallest possible template is used to reduce the dependence between pixels in the gradient calculation process while maintaining a certain degree of independence. We assume that $i(x, y)$ is the image gray value at pixel (x, y); the gradient calculation formula is as follows:

$$g_x(x, y) = \frac{i(x+1, y) + i(x+1, y+1) - i(x, y) - i(x, y+1)}{2},$$
$$g_y(x, y) = \frac{i(x, y+1) + i(x+1, y+1) - i(x, y) - i(x+1, y)}{2}. \quad (1)$$

The gradient angle calculation formula is as follows:

$$\arctan\left(\frac{g_x(x, y)}{-g_y(x, y)}\right). \quad (2)$$

The gradient amplitude calculation formula is as follows [20]:

$$G(x, y) = \sqrt{g_x^2(x, y) + g_y^2(x, y)}. \quad (3)$$

(3) The algorithm uses a greedy algorithm to pseudosort the gradient magnitudes calculated in the second step. If a normal sorting algorithm processes n data, the time complexity of pseudosorting is linear, which can save time to a certain extent. Pseudosorting is to divide the obtained gradient amplitude range (0–255) into 1024 levels, each gradient amplitude is divided into a level, and the same gradient amplitude is divided into the same level. At the same time, a state table is established, all pixels are set to UNUSED, and then the state corresponding to the pixels whose gradient amplitude is less than is set to USED. Among them, there are the following:

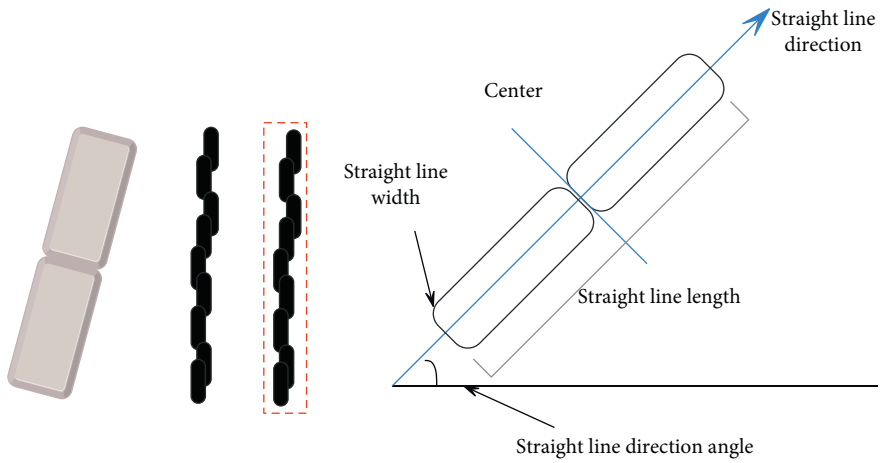

Figure 4: Rectangular approximate schematic diagram of the line segment support area.

$$\rho = \frac{q}{\sin \tau}. \tag{4}$$

In the above formula, q represents the error boundary that may occur in the gradient quantization process. According to the empirical value, q is set to 2. τ represents the angle tolerance in the fourth step of the area growth algorithm and is usually set to 22.5°.

(4) The algorithm uses the area growth algorithm to generate the line segment support area, as shown in Figure 3. The algorithm first takes the pixel with the largest gradient amplitude as the seed point (we usually think that the higher the gradient amplitude, the stronger the edge) and then searches for the pixel with the state of UNUSED in the neighborhood of the seed point. If the absolute value of the difference between the gradient angle and the area angle is between 0-τ, the pixel is added to the area. Here, the initial area angle is the gradient angle of the seed point. Each time a new pixel is added to the area, the area angle needs to be updated.

The formula for updating the area angle is as follows:

$$\arctan\left(\frac{\sum_j \sin(\theta_j)}{\sum_j \cos(\theta_j)}\right). \tag{5}$$

Among them, θ_j represents the gradient angle of the pixel j in the area, and then the process is repeated until no pixels can be added to the area.

(5) The algorithm estimates the rectangle of the line segment support area calculated in the fourth step. The result of the fourth step calculation is a series of adjacent discrete points; therefore, they need to be contained in a rectangular box r (the rectangle is the candidate of the line segment), as shown in Figure 4. The size of the rectangle is mainly selected to cover the entire area, that is, the smallest rectangle that can contain the area supported by the line segment. Obviously, this rectangular frame r contains not only the points in the line segment support area, which are also called alignment points but also includes the points close to the line segment support area, which do not belong to the outer points. The center coordinates of the rectangle are as follows [21]:

$$c_x = \frac{\sum_{jer} G(j) \cdot x(j)}{\sum_{j\epsilon r} G(j)},$$
$$c_y = \frac{\sum_{jer} G(j) \cdot y(j)}{\sum_{j\epsilon r} G(j)}. \tag{6}$$

Among them, $G(j)$ is the gradient magnitude of the pixel j, and the main direction of the rectangle is set as the angle of the eigenvector corresponding to the smallest eigenvalue of the matrix.

(6) The algorithm verifies whether the candidate rectangle is a straight line segment by calculating the Number of False Alarms (NFA). The calculation formula of NFA is as follows:

$$\mathrm{NFA} = N_{tet} \cdot P_{H_0}[k(r, I) \geq k(r, i)]. \tag{7}$$

Here, N_{test} refers to the number of potential rectangular boxes with an image size of M^*N:

$$N_{\mathrm{test}} = (NM)^{5/2}. \tag{8}$$

Among them, $k(r, I)$ is the number of alignment points in the rectangle r in the contrast model (refers to the hypothetical perfect noise image model, the characteristic of this model is that the gradient angles are randomly distributed independently and evenly distributed in $[0 \sim 2\pi]$), and $k(r, i)$ is the number of alignment points in the rectangle r at the same position in the image i to be detected. Here, the false alarm number NFA represents the probability that the number of alignment points in a certain candidate rectangle r in the image to be detected is less than the number of alignment points in the same position r in the control model. The larger the NFA is, the more similar the current r is to the same position r in the control model, and the less likely it is

to be a straight line target for detection; on the contrary, the more likely it is to be a straight line. We also know the following [22]:

$$P_{H_0}[k(r,I) \geq k(r,i)] = B(n(r), k(r,i), p), \tag{9}$$

$B(n,k,p)$ represents the binomial distribution, as shown below:

$$B(n,k,p) = \sum_{j=k}^{n} \binom{n}{k} p^j (1-p)^{n-j}, \tag{10}$$

n represents the total number of pixels in the rectangle r, k represents the number of alignment points in the rectangle r in the image i to be detected, and p represents the probability that the pixel points in the control model are aligned points, as shown below [23]:

$$p = \frac{\tau}{\pi}. \tag{11}$$

Therefore, the NFA of a matrix r is finally obtained as follows:

$$\mathrm{NFA}(r) = (NM)^{5/2} \cdot \sum_{j=k}^{n} \binom{n}{k} p^j (1-p)^{n-j}. \tag{12}$$

If $\mathrm{NFA}(r) \leq \varepsilon$, the rectangular area is considered to be a straight line, which ε is set to 1. Here, the threshold can be changed, and there is no significant difference in the detection results, so we uniformly use the threshold value of 1.

In the LSD algorithm, we assume the angle threshold δ of the line segment L, then the NFA of the line segment L is defined as follows:

$$\mathrm{NFA}(L) = \chi N B(n,k,p) = \chi N \sum_{x=k}^{n} \binom{n}{x} p^x (1-p)^{n-x}. \tag{13}$$

Among them, χ is a normalized value, N is the number of potential line segments in the image to be detected, B is the binomial distribution, n is the total number of pixels in the line segment L, k represents the number of alignment points in the line segment L, and p refers to the probability that a random pixel is an alignment point. When and only if $\mathrm{NFA}(L)$ is less than a given threshold, the line segment L is considered to be meaningful.

On the basis of the LSD algorithm, the algorithm adds scale information; that is, the algorithm first finds a longer line segment on a coarse scale and then further refines its position on a finer scale. At each scale, new line segment candidates are still considered for the same position instead of just using the line segment detected by the previous scale and then using the multiline segment criterion to merge these candidate line segments. The original image is denoted as I, and the length and width of the image are reduced by $1/2^n$, respectively, as shown below:

$$I^n = \frac{I}{2^n}, \quad n = 0, 1, 2. \tag{14}$$

I represents the finest scale, which is the original image. The larger the $n$the thicker the scale. The maximum value of n depends on whether the length or width of the I^n image is one of the pixel values below 500. If it is lower than 500, the scale is no longer reduced. The algorithm first uses the LSD algorithm to detect line segments on the coarse scale. We assume that the line segment L^n is detected in the coarse-scale image I^n, the line segment direction is denoted as $\theta(L^n)$, and the given angle threshold is δ^n. We define S^{n-1} as the rectangular area corresponding to L^n enlarged in the fine-scale image I^{n-1}. We denote R^{n-1} as a subset of pixels in S^{n-1} where the gradient direction of the pixel and the angle difference between the direction of the line segment L^n are lower than the threshold δ^n, as shown below:

$$R^{n-1} = \left\{ q \in S^{n-1} \text{and} \left| \theta(q) - \theta\left(L^n\right) \right| < \delta^n \right\}. \tag{15}$$

Then, the algorithm calculates a set E^{n-1}, which includes all the connected components in R^{n-1}, thus generating potential new line segments. These components may belong to the same line segment, may be parallel to each other, or close to the same line segment. They are fused together in the coarse-scale image and are judged as a line segment.

We assume that n line segments $S = \{s_1, \ldots, s_n\}$ are given, and $\$L\$$ is the best line segment calculated from the line segment set s_i. The rectangle corresponding to this line segment refers to the smallest rectangle that contains the rectangles associated with all line segments s_i. The corresponding fusion score of this group of line segments is defined as follows:

$$A(s_1, \ldots, s_n) = \log\left(\frac{\mathrm{NFA}_M(s_1, \ldots, s_n, p)}{\mathrm{NFA}_M(L, p)} \right). \tag{16}$$

If the fusion score is positive, it means that NFA_M is lower than the NFA of a single line segment, so it should be merged. This defines a line segment merging standard that does not depend on any parameters and thus has an adaptive characteristic.

In fact, the set E^{n-1} contains many temporary line segments, and there are countless combinations of these temporary line segments, which we cannot test one by one. Therefore, it can be iterated by a greedy algorithm. The algorithm first selects the smallest NFA component e from the set $E^{n-1}$and then uses e as the benchmark to calculate all other components that are fully aligned with it, as shown below:

$$C(e) = \left\{ e' \in E^{n-1} \backslash \{e\} \text{and} l(e) \cap l\left(e'\right) \neq \varnothing \right\}. \tag{17}$$

Among them, $l(e)$ is a line segment passing through the center of component e, the angle is $\theta(e)$, $l(e')$ is a line segment passing through the center of component e', and the angle is $\theta(e')$. Then, the algorithm calculates the fusion score calculation method of $C(e)$ as shown in equation (16). If it is positive, the algorithm replaces the subline segments in $C(e)$ with the merged version of $C(e)$. It continues to iterate until all the temporary segments in E^{n-1} have been tested.

Finally, the algorithm calculates the NFA of all the line segments, leaving only the meaningful ones. When there is noise or the contrast is low, the line segment L^n detected in

the coarse-scale image usually does not meet the NFA condition. Therefore, it is often impossible to derive the line segment in the fine-scale image. In this case, the original line segment in the coarse scale is directly retained, and no more attempts are made to find a finer line segment at the same position in the fine-scale image, and the line segment is extracted only at this scale using LSD.

The algorithm first matches all the line segments in L_i with all the line segments in L_j to generate a potential corresponding relationship, where L_i refers to the line segment detected in image I_i, L_j refers to the line segment detected in image I_j, and I_i and I_j are two images to be matched. For specific line segment pairs, $l_m^i \in L_i, l_m^j \in L_j$. The algorithm first calculates the epipolar lines corresponding to the end points of the line segment, such as the extreme lines of the end points q_m^i and p_m^i of l_m^i in the image I_j. The algorithm intersects $l_{\overline{m}}^j$ and these two polar lines and obtains two intersection points x_p and x_q, and the two end points $p_{\overline{m}}^j$ and V of x_p and x_q are collinear with $l_{\overline{m}}^j$, and then defines a matching score between the line segments Z and X, as shown below:

$$s\left(l_m^i, l_{\overline{m}}^j\right) = \frac{\text{inner}\left(\left\{p_{\overline{m}}^j, q_{\overline{m}}^j, x_p, x_q\right\}\right)}{\text{outer}\left(\left\{p_{\overline{m}}^j, q_{\overline{m}}^j, x_p, x_q\right\}\right)}. \tag{18}$$

Among them, inner ({...}) and outer ({...}), respectively, represent the Euclidean distance corresponding to a pair of outer points and a pair of inner points among the four collinear points. (18) illustrates the degree of matching between two two-dimensional line segments. If all the line segments can be detected ideally, that is, no occlusion, not too long, and not too short, then the matching score $s(l_m^i, l_{\overline{m}}^j)$ will be exactly 1. Therefore, in general, if the matching score is greater than a fixed threshold, it is considered that there is a potential matching correspondence between the line segments $l_{\overline{m}}^j$ and l_m^i.

Knowing all the camera poses, we can use the camera pose to project each two-dimensional corresponding relationship into the three-dimensional space to get a three-dimensional line segment hypothesis. For example, the algorithm transforms the two-dimensional correspondence relationship $l_m^i \longrightarrow l_{\overline{m}}^j$ to the three-dimensional line segment $H_{m,\overline{m}}^{i,j}$, where $H_{m,i}^{i,j}$ is the intersection of the plane formed by the camera center $C_i, C_j \in R^3$ and the two-dimensional line segments l_m^i and $l_{\overline{m}}^j$. For each corresponding relationship, calculate two three-dimensional line segment hypotheses ($h_{m,m}^{i,j}$and$h_{m,m}^{j,i}$), respectively, and they all fall on the three-dimensional line segment $H_{m,\overline{m}}^{i,j}$, where the end points of the back projection of $h_{m,i}^{i,j}$ and $h_{m,m}^{j,i}$ coincide with the end points of l_m^i and l_m^j, respectively. Similar to a two-dimensional line segment, a three-dimensional line segment is also composed of two three-dimensional end points $h_{m,m}^{i,j} = \left\{P_{m,\overline{m}}^{i,j}, Q_{m,\overline{m}}^{i,j}\right\}$. Note that $H_{m,\overline{m}}^{i,j} = H_{m,m}^{j,i}$. Usually, due to occlusion and inaccurate two-dimensional line segment detection, $h_{m,\overline{m}}^{i,j} \neq h_{\overline{m},m}^{j,i}$. However, the line segments $h_{m,\overline{m}}^{i,j}$ and $h_{\overline{m},m}^{j,i}$ are always collinear with the infinite line segment $H_{m,\overline{m}}^{i,j}$. Then, by analyzing the spatial consistency of the three-dimensional line segment hypothesis,

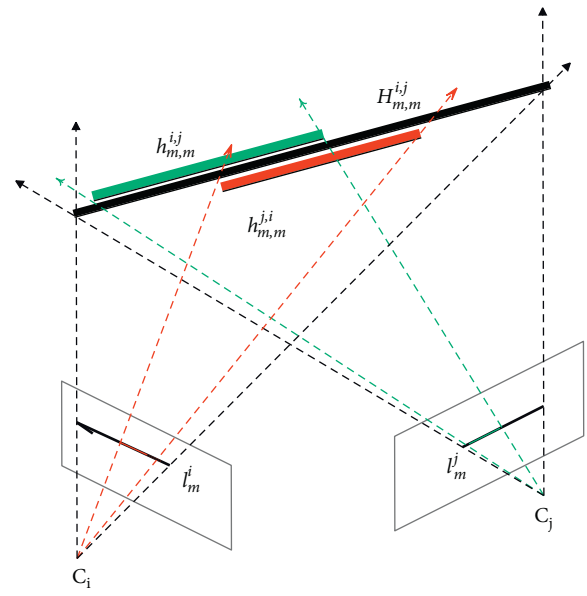

Figure 5: The hypothesis of the three-dimensional line segment generated by the two-dimensional potential matching line segment.

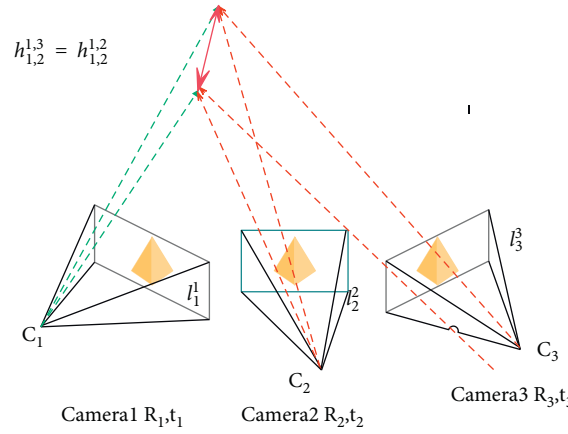

Figure 6: Pairwise correct matching results in a spatially consistent three-dimensional line segment hypothesis.

the correct match can be selected from the incorrect matching line segments. The hypothesis of the three-dimensional line segment generated by the two-dimensional potential matching line segment is shown in Figure 5.

Next, we need to calculate the confidence of each pair of matching line segments between all images and all of their neighbors. The algorithm judges whether the three-dimensional hypothesis $h_{m,i}^{i,j}$ generated by the line segment l_m^i and the line segment l_m^j is reasonable through the confidence, that is, whether the three-dimensional hypothesis is suitable for all matching images (except for the matching of I_i and I_j). We assume that the line segment l_m^i in image I_i and the line segment l_m^j in image I_j have been correctly matched, and the other line segment l_m^b in image I_b has been correctly matched, then the 3D hypotheses $h_{m,\overline{m}}^{i,j}$ calculated from lines l_m^i and l_i^j and the 3D hypotheses $h_{m,i}^{i,b}$ calculated from l_m^i and l_m^i should be very close to each other in space as

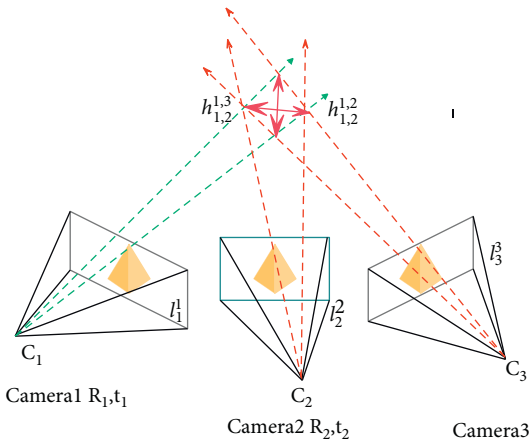

FIGURE 7: Pairwise incorrect matching results in a spatially inconsistent three-dimensional line segment hypothesis.

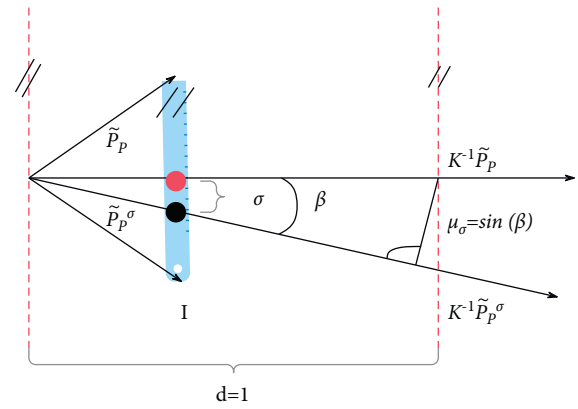

FIGURE 8: Scale regularization.

shown in Figure 6 (in the ideal noise-free case, they should be perfectly colinear). On the contrary, if the matching is incorrect, the three-dimensional hypothesis obtained cannot be spatially close to that shown in Figure 7. This is because the wrong assumptions obtained by triangulation are not geometrically consistent. However, the correct assumptions obtained by triangulation always support each other. Therefore, this feature of geometric consistency can be used to eliminate mismatches.

To measure the similarity based on the spatial distance and angular error between two three-dimensional hypotheses, we first define confidence for a corresponding relationship $h^{i,j}_{m,\overline{m}}$, as shown below:

$$c\left(h^{i,j}_{m,\overline{m}}\right) = \sum_{q \in V^M_i \setminus \{j\},} \max_{p \in \{1,\ldots,m_x\}} \left\{A\left(h^{i,j}_{m,\overline{m}}, h^{i,q}_{m,p}\right)\right\}. \tag{19}$$

Among them, the A calculation is derived from the correlation between two three-dimensional hypotheses of the same two-dimensional line segment l^i_m. This correlation is defined as follows:

$$A\left(h^{i,j}_{m,\overline{m}}, h^{i,q}_{m,p}\right) = \begin{cases} \min\left\{S^a\left(h^{i,j}_{m,j}, h^{i,q}_{m,p}\right), S^d\left(h^{i,j}_{m,j}, h^{i,q}_{m,p}\right)\right\}, & \text{if } \min\{\cdots\} > \frac{1}{2}, \\ 0, & \text{else.} \end{cases} \tag{20}$$

S^a is the angle similarity in the three-dimensional line segment hypothesis, and $S(d)$ is the position similarity in the three-dimensional line segment hypothesis, which is defined as follows:

$$S^a\left(h^{i,j}_{m,\overline{m}}, h^{i,q}_{m,p}\right) = \exp\left(-\frac{\angle\left(h^{i,j}_{m,\overline{m}}, h^{i,q}_{m,p}\right)^2}{2\sigma_a^2}\right),$$

$$S^d\left(h^{i,j}_{m,\overline{m}}, h^{i,q}_{m,p}\right) = \min_{Z \in h^h_{j,m},m} \exp\left(-\frac{l_\perp\left(Z, h^{i,q}_{m,p}\right)^2}{\sigma_x\left(l_i(Z)\right)^2 + \sigma_y\left(l_i(Z)\right)^2}\right). \tag{21}$$

Among them, $\angle(h_1, h_2)$ represents the angle between two line segments (in degrees), $l_\perp(Z, h_2)$ is the vertical distance between the three-dimensional point Z and the straight line passing through h_2, and $l_i(Z) = \|C_i - Z\|_2$ is the Euclidean distance between the camera center C_i of the image I_i and the three-dimensional point Z, that is, the depth of the three-dimensional point Z along the optical axis of the camera. In order to prevent only a few weak supporters, the confidence is also high. We cut the correlation and only accept the values above 1/2.

Using the depth-adaptive spatial regularization function $\sigma_p(d_i(Z))$[72], this regularization function is defined as follows:

$$\sigma_P(d) = d \cdot \frac{\mu}{d_{\text{med}}}. \tag{22}$$

This is a linear function of depth d. The slope μ/d_{med} of this function is composed of the specified spatial regularization factor μ (for example, 5 cm of the reconstruction result corresponds to 0.05) and the regularization depth d_{med}. In this paper, d_{med} simply refers to the distance from the world point to the camera. However, this formula needs to know the reconstruction scale information in advance (the decision of μ requires scale information). The fact is that the size information of obstacles is often not known. Therefore, a formula with a constant scale is used to deal with this situation, as shown below:

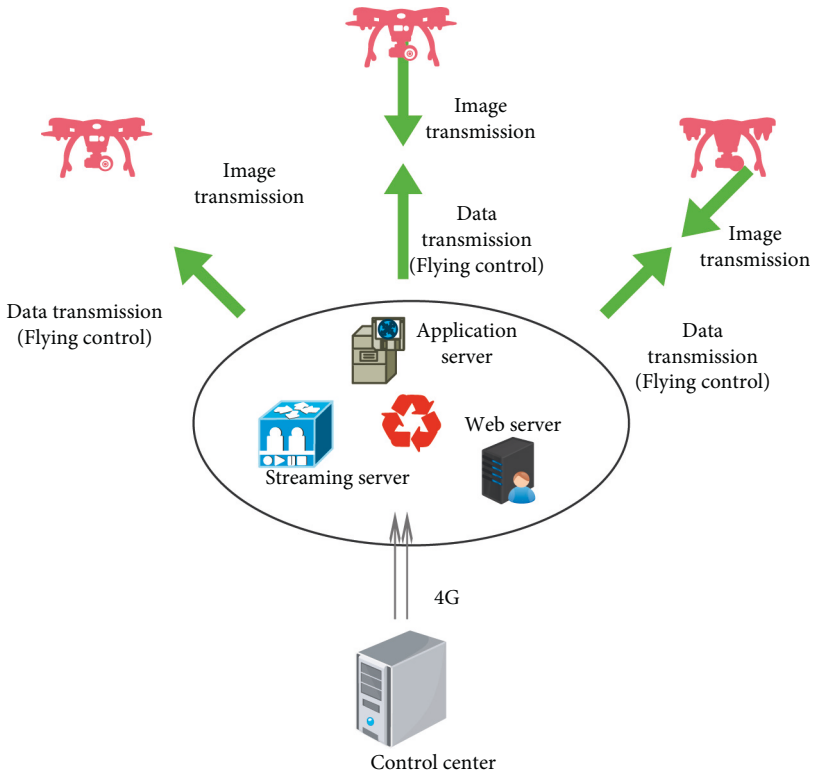

Figure 9: The flow of the UAV intelligent control system based on machine vision and multiagent decision-making.

$$\sigma_p^i(d) = d \cdot \mu_\sigma. \tag{23}$$

This is also a linear function of depth d. However, this time the slope μ_σ is derived from the geometric mechanism of the camera. We assume that given a standard pinhole camera model, move the origin $\widetilde{p}_p$ horizontally by a regularized σ pixel to obtain $\widetilde{p}_p^\sigma$ ($\sim$ represents homogeneous coordinates), and then calculate the angle β between the two three-dimensional rays $K^{-1} \cdot \widetilde{p}_p$ and $K^{-1} \cdot \widetilde{p}_p^\sigma$, where K^{-1} is the internal parameter matrix of the camera.

Then, the algorithm simply calculates $\mu_\sigma = \sin(\beta)$, which is basically the maximum distance to move the origin of the camera at depth $d = 1$, so that the distance between the reprojection of the moving point and the midpoint of the image is less than or equal to σ, as shown in Figure 8. This formula ensures that when the corresponding 3D line segment is assumed to be far away from the camera, the greater the distance from the 3D point to the line segment, the less the penalty, and vice versa. Therefore, in order to maintain scale invariance, the new $\sigma_p^i(d)$ is used instead of $\sigma_p(d)$ here.

It is now possible to determine whether a matching three-dimensional line segment hypothesis makes sense. Only when $c(h_{m,\overline{m}}^{i,j}) > 1$, this assumption is retained for further processing, which means that at least two line segments from two additional images (except I_i and I_j) support $h_{m,\overline{m}}^{i,j}$. Therefore, a set of sparser correspondences can be finally obtained, and most of the mismatches are removed.

4. UAV Intelligent Control Based on Machine Vision and Multiagent Decision-Making

The intelligent control process of UAV based on machine vision and multiagent decision-making is shown in Figure 9. When traditional UAVs operate remotely, they need to carry ground station equipment. Usually, a ground station device can only control one drone at a time, and the ground station and the command center can transmit data through the 5G network. When UAV operations are carried out through the intelligent control of drones based on machine vision and multiagent decision-making, there is no need to carry special ground station equipment, and the functions of the ground station software are deployed on the cloud platform. This effectively reduces the cost of system hardware while obtaining the powerful computing power of cloud computing. In addition, different from the one-to-one matching of UAVs and ground stations in the traditional way, in the UAV intelligent control system based on machine vision and multiagent decision-making, each UAV is a node of the system, and the UAV can be identified through the network address, and multiple UAVs can be controlled at the same time.

In order to improve the efficiency of multiagent decision-making, this paper proposes to carry out intelligent control data transmission and processing on the cloud platform. The UAV cloud control scheme proposed in this paper is shown in Figure 10, which mainly includes three parts: terminal equipment, cloud platform, and UAV.

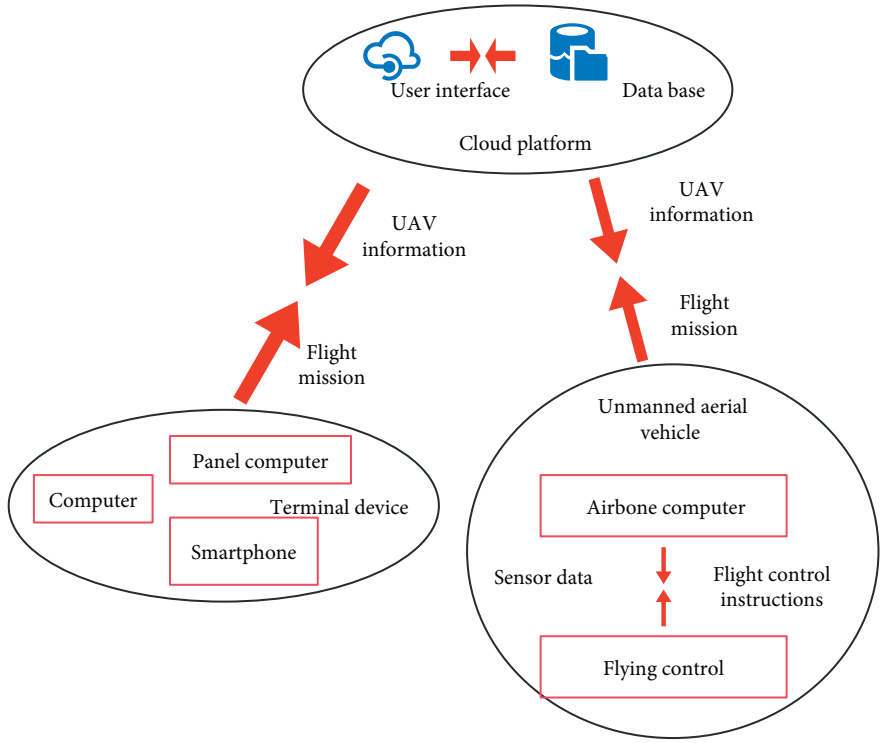

Figure 10: Schematic diagram of the scheme.

Table 1: Machine vision effect detection.

Number	Machine vision
1	93.574
2	95.891
3	93.206
4	97.714
5	93.016
6	96.640
7	96.378
8	96.080
9	93.348
10	96.169
11	97.773
12	94.359
13	93.397
14	96.687
15	95.397
16	95.699
17	95.067
18	93.170
19	93.181
20	95.418
21	96.714
22	96.528
23	95.344
24	97.915
25	96.423
26	95.724
27	95.602
28	96.282
29	93.132

Table 1: Continued.

Number	Machine vision
30	93.033
31	93.850
32	93.065
33	95.412
34	94.642
35	96.910
36	96.705
37	97.502
38	97.215
39	94.618
40	97.666
41	97.794
42	97.408
43	96.531
44	97.147
45	94.713
46	96.370
47	93.667
48	94.060
49	95.951
50	96.590
51	97.032
52	97.511
53	97.285
54	93.291
55	96.034
56	95.009
57	97.524

Table 2: The control effect of the UAV intelligent control system based on machine vision and multiagent decision-making.

Number	Control effect
1	91.992
2	93.169
3	89.641
4	89.259
5	90.691
6	88.784
7	93.029
8	88.650
9	93.561
10	92.127
11	91.382
12	91.923
13	89.110
14	89.091
15	93.236
16	88.166
17	93.668
18	91.834
19	93.654
20	91.288
21	88.472
22	89.224
23	89.139
24	90.184
25	88.419
26	89.019
27	92.395
28	93.662
29	88.304
30	89.978
31	92.967
32	89.431
33	88.081
34	88.010
35	92.612
36	93.456
37	90.242
38	90.020
39	89.008
40	93.271
41	90.150
42	91.875
43	91.077
44	93.118
45	90.849
46	91.347
47	90.163
48	88.486
49	93.761
50	90.835
51	92.676
52	90.935
53	93.267
54	91.940
55	92.074
56	91.276
57	92.169

After constructing the above model, the model of this paper is tested and researched. The model in this paper uses machine vision for intelligent recognition and performs intelligent control of UAVs under the support of multiagent decision-making. Therefore, this paper uses intelligent machine vision to recognize UAV images, and the results are shown in Table 1.

Based on the above detection, it can be seen that the machine vision method proposed in this paper performs better in UAV visual recognition. On this basis, the intelligent control effect of UAVs based on machine vision and multiagent decision-making can be verified, and the results shown in Table 2 below are obtained.

From the above research, it can be seen that the UAV intelligent control system based on machine vision and multiagent decision-making can achieve reliable control of UAVs and improve the work efficiency of UAVs.

5. Conclusion

The UAV flight control system is developed on the basis of manned aircraft, but in contrast, there are some new technical requirements. The primary function of the UAV flight control system is to enable the UAV to autonomously control the flight attitude, flight speed, and flight path. At the same time, the UAV flight control system needs to send a series of instructions to dispatch the various functional components of the aircraft during the flight of the aircraft, receive feedback information, and vote on redundant subsystems. Finally, when the aircraft system fails, the flight control system must have the ability to self-check the failure and restore the aircraft to normal flight through the aircraft redundancy system. This paper combines machine vision technology and multiagent decision-making technology to study the intelligent control of drones, uses intelligent machine vision technology for recognition, and uses multiagent decision-making technology for motion control to improve the motion effect of drones. The research shows that the UAV intelligent control system based on machine vision and multiagent decision-making proposed in this paper can achieve reliable control of UAVs and improve the work efficiency of UAVs.

Data Availability

The labeled dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was sponsored by the Hubei University of Technology.

References

[1] F. L. Duarte and R. C. De Lamare, "C-RAN-Type chd relaying with recursive maximum minimum distance," *IEEE Communications Letters*, vol. 24, no. 11, pp. 2623–2627, 2020.

[2] L. Hu, Y. Tian, J. Yang, T. Taleb, L. Xiang, and Y. Hao, "Ready player one: UAV-Clustering-Based multi-task offloading for vehicular VR/AR gaming," *IEEE Network*, vol. 33, no. 3, pp. 42–48, 2019.

[3] W. Yi, Y. Liu, E. Bodanese, A. Nallanathan, and G. K. Karagiannidis, "A unified spatial framework for UAV-a networks," *IEEE Transactions on Communications*, vol. 67, no. 12, pp. 8801–8817, 2019.

[4] W. Feng, J. Wang, Y. Chen, X. Wang, N. Ge, and J. Lu, "UAV-aided MIMO cit," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1731–1740, 2019.

[5] H. He, S. Zhang, Y. Zeng, and R. Zhang, "Joint altitude and bo-enabled multiuser communications," *IEEE Communications Letters*, vol. 22, no. 2, pp. 344–347, 2018.

[6] B. Li, S. Zhang, X. Zhang, B. Xi, and Y. Tian, "Application and research of scheduling mechanism for UAV cluster launching control system," *Xibei Gongye Daxue Xuebao/Journal of Northwestern Polytechnical University*, vol. 36, no. 2, pp. 353–358, 2018.

[7] W. Mei, Q. Wu, and R. Zhang, "Cellular-connected UAV: uplink association, power control and interference coordination," *IEEE Transactions on Wireless Communications*, vol. 18, no. 11, pp. 5380–5393, 2019.

[8] Q. Zhu, K. Jiang, X. Chen, W. Zhong, and Y. Yang, "A novel 3D non-stationary UAV-MIMO channel model and its statistical properties," *China Communications*, vol. 15, no. 12, pp. 147–158, 2018.

[9] X. Wang and M. C. Gursoy, "Coverage analysis for energy-harvesting UAV-a cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 12, pp. 2832–2850, 2019.

[10] M. Y. Arafat and S. Moh, "Localization and clustering based on swarm intelligence in UAV networks for emergency communications," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8958–8976, 2019.

[11] D. Ebrahimi, S. Sharafeddine, P. H. Ho, and C. Assi, "UAV-aided projection-based compressive data gathering in wireless sensor networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1893–1905, 2019.

[12] S. Fu, Y. Tang, N. Zhang, L. Zhao, S. Wu, and X. Jian, "Joint unmanned aerial vehicle (UAV) deployment and power control for internet of things networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4367–4378, 2020.

[13] D. Liu, Y. Xu, J. Wang et al., "Opportunistic UAV utilization in wireless networks: motivations, applications, and challenges," *IEEE Communications Magazine*, vol. 58, no. 5, pp. 62–68, 2020.

[14] X. Liu, Y. Liu, and Y. Chen, "Reinforcement learning in multiple-UAV networks: deployment and movement design," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 8036–8049, 2019.

[15] T. Yu, X. Wang, and A. Shami, "UAV-enabled spatial data sampling in large-scale IoT systems using da neural network," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1856–1865, 2019.

[16] Y. S. Wang, Y. W. P. Hong, and W. T. Chen, "Trajectory learning, clustering, and user association for dynamically connectable UAV base stations," *IEEE Transactions on Green Communications and Networking*, vol. 4, no. 4, pp. 1091–1105, 2020.

[17] M. Khabbaz, J. Antoun, and C. Assi, "Modeling and performance analysis of UAV-assisted vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8384–8396, 2019.

[18] D. Liu, J. Wang, K. Xu et al., "Task-driven relay assignment in distributed UAV communication networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, Article ID 11003, 2019.

[19] W. Rahmaniar and A. E. Rakhmania, "Online digital image stabilization for an unmanned aerial vehicle (UAV)," *Journal of Robotics and Control (JRC)*, vol. 2, no. 4, pp. 234–239, 2021.

[20] A. Thibbotuwawa, G. Bocewicz, P. Nielsen, and B. Zbigniew, "Planning deliveries with UAV routing under weather forecast and energy consumption constraints," *IFAC-PapersOnLine*, vol. 52, no. 13, pp. 820–825, 2019.

[21] N. H. Motlagh, M. Bagaa, and T. Taleb, "UAV-based IoT platform: a crowd surveillance use case," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, 2017.

[22] X. Yu, X. Dong, X. Yang et al., "Air–ground integrated deployment for UAV enabled mobile edge computing: a hierarchical game approach," *IET Communications*, vol. 14, no. 15, pp. 2491–2499, 2020.

[23] X. Liu, Y. Liu, Y. Chen, and L. Hanzo, "Trajectory design and power control for multi-UAV assisted wireless networks: a machine learning approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7957–7969, 2019.