IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 September 2022, accepted 19 September 2022, date of publication 22 September 2022,
date of current version 29 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3208685

RESEARCH ARTICLE

An AI-in-Loop Fuzzy-Control Technique for UAV's Stabilization and Landing

MOHAMMED RABAH[1,2], HASHEM HAGHBAYAN[3], (Member, IEEE), EERO IMMONEN[1], AND JUHA PLOSILA[3], (Member, IEEE)

[1]Computational Engineering and Analysis Research Group, Turku University of Applied Sciences, 20520 Turku, Finland
[2]Department of Communications Engineering, Al-Safwa High Institute of Engineering, Cairo 11837, Egypt
[3]Department of Computing, University of Turku, 20500 Turku, Finland

Corresponding author: Mohammed Rabah (mohamed.rabah@turkuamk.fi)

ABSTRACT In this paper, an adaptable fuzzy control mechanism for an Unmanned Aerial Vehicle (UAV) to manipulate its mechanical actuators is provided. The mission (landing) for the UAV is defined to track (land on) an object that is detected by a deep learning object detection algorithm. The inputs of the controller are the location and speed of the UAV that have been calculated based on the location of the detected object. Two separate fuzzy controllers are proposed to control the UAV's motor throttle and its roll and pitch over the mission and landing time. Fuzzy logic controller (FLC) is an intelligent controller that can be used to compensate for the non-linearity behaviour of the UAV by designing a specific fuzzy rule base. These rules will be utilized to adjust the control parameters during the mission and landing period in runtime. To add the effect of the ground for tuning the FLC membership function over the landing operation, a computational flow dynamic (CFD) modeling has been investigated. The proposed techniques is evaluated on MATLAB/Simulink simulation platform and real environment. Statistical analysis of the UAV location reported during stabilization and landing process, on both simulation and real platform, show that the proposed technique outperforms the similar state-of-art control techniques for both mission and landing control.

INDEX TERMS Fuzzy control, fuzzy-PID, UAV, ground effect, stabilization and landing.

I. INTRODUCTION

Recent stabilization and landing algorithms for UAVs have been utilizing the advancements in the field of deep learning and artificial intelligence [1], [2]. Even though development of systems with deep learning and control has been widely studied for UAVs, battery limitation of the UAV from one side and need of fast and accurate UAV's reaction in different environmental noise and during the mission and landing period is still a challenge [3].

For a smooth, fast, and reliable UAV movement, regardless of the perceptual sensors that has been used to control it, such as GPS [4], optical flow [5], ultrasonic sensors [6] and laser range finder [7], the controller should manipulate the actuators in different environmental situations during the mission and landing periods. These environmental conditions are accompanied by noise and in some cases, such as the period of landing the UAV, affects the dynamics of the UAV and must be considered in control design in runtime. On top of this all the computing and mechanical process demand onboard power resource that, in the case of UAVs, is the mounted battery. Even though there exists several control techniques for UAV controls, still there exists a unified lightweigh control technique that can be adopted based on different environmental conditions in runtime.

Another issue is the perceptual sensors to detect the suitable landing place that might be even mobile. Using GPS data to detect the landing place removes significant portion of onboard power consumption for object detection [4]; however it is noisy and limits the operation of the UAV on to those areas that GPS is available [8]. Infrared based-approach allow to estimate the pose of a UAV by using infrared lamps and

The associate editor coordinating the review of this manuscript and approving it for publication was Zhongyi Guo.

placing it in the landing area of the UAV [9]; nevertheless, this technique is limited to only this environment. Vision-based control techniques uses onboard sensors, e.g., camera, and utilize the information obtained from the sensor to fed it to the feedback control algorithm, which is responsible for stabilizing and landing the UAV. Such technique can be executed onboard, and thus overcome the limitation of the previous techniques.

During an autonomous application, landing is considered the most critical part. To land a UAV safely, a ground effect which exists near to the ground should be carefully considered. Generally, ground effect has a nonlinear property which can't be dealt with ordinary controllers. Therefore, In this paper, an adaptable fuzzy control mechanism for the UAV to manipulate its actuators is proposed, i.e., throttle adjustment and its roll and pitch, during the mission and landing period. FLC is an intelligent controller which can effectively compensate for the nonlinear ground effect by designing specific rule base. For this, two separate fuzzy controller are proposed: 1) to adjust the UAV's throttle based on two inputs of speed and distance from the ground; and 2) to *adjust* the parameters of the PID controller based on two observations of position of the UAV and speed. The former controller is responsible to adjust the throttle of the UAV to provide fast and smooth navigation movement while the latter manipulates the roll and pitch of the UAV for efficient stabilization in different environmental conditions. A well-known Mamdani and Assilian [10] fuzzy inference system to synthesize a set of linguistic control rules that is defined in design time during the landing and normal operation of the UAV is utilized. The membership functions of the landing FLC set through comprehensive study of modeling the effect of the ground on the propellers. The proposed technique has been evaluated on both simulation and real environment platform. For the simulation platform, MATLAB/Simulink has been used, under the effect of random disturbances in X, Y, and Z directions, and the statistical error values about the three directions were obtained. For the real platform, the proposed controller is implemented on an experimental UAV and is compared to other controllers. Tests were performed in real-time, and the statistical error values about the three directions were obtained.

The main contributions of this work can be summarized as follows:

- Developing a non-linear Fuzzy-PID controller, that is responsible for the stabilization of the UAV during the landing process.
- Developing a Fuzzy Logic Controller (FLC) that utilize the position and speed of the UAV to ensure a smooth landing especially near the ground to avoid the ground effect phenomena.
- Adjusting the parameters of the controller based on comprehensive modeling the effect of environment on the UAV by using computational flow dynamic modeling.
- Development of both simulation and real platform to evaluate the proposed idea and comparing the results with state-of-the-art.

The remaining part of this paper is divided as follows: Section II reviews the related work about the developed control systems based on external environment and internal environment. System overview is illustrated in III. Section IV explains the algorithms used in the developed system in detail. The experimental setup is covered in Section V. Simulation and real platform results are demonstrated in VI followed by the conclusion in Section VII.

II. RELATED WORK AND MOTIVATION

Recent advancements in both fields of UAV control system and artificial intelligence (AI) have made a certain realm of vision-based autonomous applications possible [11], [12], [13], [14]. The operation of a fully automated UAV system can be divided into three working periods namely 1) take off period 2) the mission period, i.e., tasks to be performed, and 2) the landing period. Among these three periods, efficient and robust control techniques for UAV stabilization over the mission and landing period is an important necessity, since the UAV must full-fill several requirements such as smooth path trajectory, high stability, and smooth and relax landing to decrease the damage on both cargo, in the case of, e.g., delivery mission, and physical parts of the UAV [15].

In [16], the authors proposed a control system for stabilizing the UAV that is following a moving object under various speeds. To do this, a Gain-Scheduled PID controller (GPID) has been adopted that refines the system's actuation based on the received feedback. Here the actuation are the motor thrust and the feedback is the location of the object under track. However, in the proposed idea, due to linear relationship of input feedback and output actuation values of the GPID controller, the system is showing a non robust behaviour while big disturbances in the input is experienced.

In [17], the authors proposed a feed-forward proportional integral (PI) stabilizer for pUAV to solve the same problem, thereby disturbance parameters directly affect the control parameters in stabilization while tracking. However, since calculating the control parameters demands heavy workload, mainly such techniques require an offloading process through which the heavy calculation of the parameters happens in stationary resources. This highly affects the response time of the controller and makes it infeasible for UAVs to operate in areas without network connection.

The popular *PX4* autopilot software which is employed in UAV applications for stabilization and position control [18] is based on two nested loops, i.e., a proportional gain, P, in the outer loop, and PID in the inner loop. The outer is responsible for controlling and stabilizing the UAV orientation angles while the inner loop is responsible for controlling the angular velocity of the UAV. However, because of the fixed gain of the proportional gain of the inner and outer controller loops, the

controller behaviour is not robust while encountering external disturbances, e.g., wind, which will require continues change in the PID gains.

In [19], the authors proposed a tracking technique based on deep learning performing a multi-regularized correlation filter-based track. Even though the proposed method can track the target with high accuracy, executing the deep learning algorithm requires continuous data transfer to a ground station to perform the heavy deep learning process, making it not suitable for a run-time applications.

Another approach to stabilize a UAV while tracking an object or a path is by utilizing a FLC. Authors in [20] studied the response of the FLC and a PID controller while tracking a trajectory using MATLAB/Simulink, while authors in [21] implemented the FLC on a physical system and tested its performance inside a restricted area. Although the proposed controller works well, it suffers from overshoot when the UAV has to change its direction.

Two commonly used techniques for autonomous landing in UAVs are altitude control techniques and a speed control techniques. In altitude control techniques, the altitude reference, i.e., the envisioned Z axis location of the UAV, is smoothly decreased over time and the control technique aims to provide a smooth reliance of the UAV location with respect to the envisioned altitude reference. To get the altitude information, a sonar sensor is usually used [22]. In speed control approach such as what is used in *ArduPilot* [23] or what is developed by the authors in [24] and [25], the goal is to preserve a constant downhill speed for the UAV until reaching the ground level.

Altitude control approach provides a secure landing, but the landing procedure is slow which does lead to high power consumption of the battery. On the other hand, the speed control technique might lead to the crash of the UAV in case the target speed is high or big disturbance, e.g., wind impulse, happens. Therefore, it is required to combine both altitude control and speed control techniques in order to achieve the best results.

Authors in [26] proposed an auto-landing technique to control the UAV based on estimation of future altitude and speed of the UAV w.r.t the landing destination while using Kalman Filter. Despite the fact that their technique is effective, Kalman Filter implementation is complex and requires large computational effort, whereas in UAVs, time is critical and the landing process should be fast to save battery life.

In [27], a landing technique based on Deep Neural Networks was developed. To test this landing technique, the UAV should always be connected to their network, and such methods are not suitable for real-time applications due to its limited network coverage area and high latency caused by data transfer, leading to a significant degrade in performance.

In conclusion, we can categorize the stabilization control techniques into light weight simple linear, i.e., PID controller, and non-linear computational heavy techniques that are mainly based on learning from experimental data. Each of the mentioned techniques has its own benefits and drawbacks.

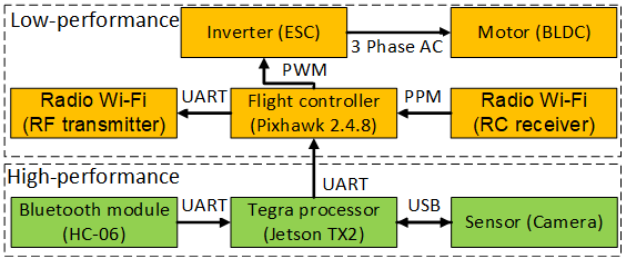


FIGURE 1. Block diagram of the overall hardware architecture system.

In light weight techniques, even though the computational cost is low and the control system is energy efficient, the non-linearity behaviour of the UAV functionality and the environmental conditions, makes the control system non-robust and untrustable, especially under the effect of high disturbances. In contrast, in heavy learning techniques, even though the non-linear behaviours of the system models is considered, run-time data processing and transfer causes huge computation and communication cost that affects the energy efficiency and response time of the control unit. Another important fact in designing such control techniques is the non-probabilistic behaviour of the UAV physics and environment that makes the possible stochastic methods practically impossible leading to a fuzzy solution for such a modeling, i.e., not-probabilistic.

Overall, in this paper a fuzzy technique is proposed to control the UAV over the mission and landing period. Generally, the FLC requires enormous computation which increase the computational burden of the system [28], [29] and to overcome this, there are some developed methods to reduce its execution time. Authors in [30] proposed a technique to simplify the FLC rule base table for induction motor drives. Their results show that the simplified rule base has similar result to the standard rule base and is less computational. Another approach to accelerate the FLC computation process is using a lookup table. Authors in [31] generated a lookup table using Octave simulation. Results show that the generated lookup table is 7 times faster than the ordinary FLC. In our work, a lookup table based FLC is used to control the UAV.

In this technique, a non-probabilistic, non-linear membership function monitors the influential parameters over the mission and landing period upon which adjusts the control parameters at run-time. In this method, there exists different modes of control each of which provides different parameters for the control system to stabilize the X-Y location over the mission period and X-Y-Z location of the UAV over the landing period. Two fuzzy controllers for the mission and landing period are proposed that are called *stabilization* and *landing* controllers respectively. The stabilization controller during the mission and landing period while the landing controller only operates during the landing period. The membership function for these two fuzzy controllers are based on different behaviors the UAV shows over the mission

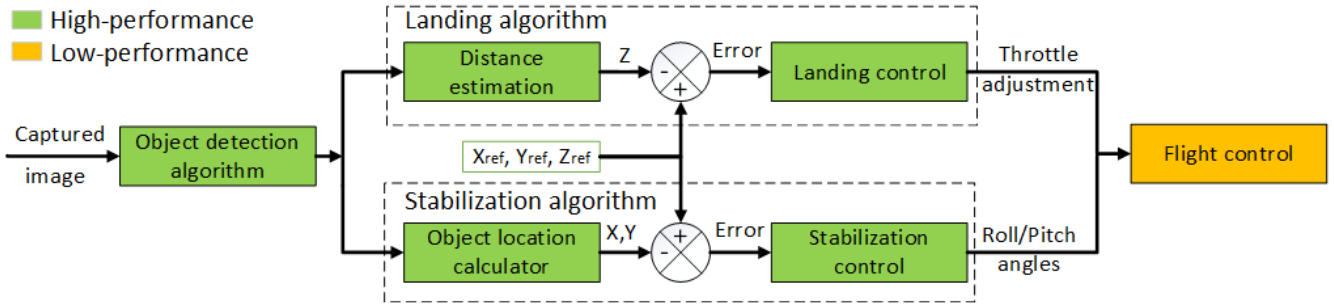


FIGURE 2. Block diagram of the software architecture of the vision-based controller.

and landing period. These functions are designed based on CFD simulation predictions of the ground effect strength for the proposed propeller configuration at different flight altitudes. Similar simulation methods have been utilized before (see e.g., [32], [33]), with the recent advances focusing on predicting propeller air flow near obstacles [34] and air flows associated with realistic UAV flight patterns [35], [36]. For UAV control system design, CFD methods have been utilized for validating the propeller ceiling effect in bridge inspection [37]. In the present work, CFD predictions are utilized for tuning the fuzzy membership functions for near-ground and far-ground conditions, for efficient stabilization and landing.

III. SYSTEM OVERVIEW

Figure 1 shows the block diagram of the overall hardware architecture of the proposed system. The architecture is divided into two main parts, i.e., high-performance and low-performance parts. The high performance part consists of a high-performance processor, Jetson TX2 in our case, high-volume data image sensor, and communication units. The received image data from the camera is processed in the Jetson TX2 for performing object detection. After detecting the object, position of the detected object and the distance of the UAV to the object are calculated and sent to the flight controller in low-performance part to calculate the best commands for stabilizing and landing. These commands will be sent to the electronic speed controller (ESC) that is responsible for controlling and regulating the speed of the UAV's brushless DC (BLDC) motors, i.e., propeller electric motors.

The low-performance part consists of a small flight controller, Pixhawk 2.4.8 in our case, and interface units. The Radio Control (RC) receiver, receives signals from the transmitter and send it to the flight controller, which puts this information into action by controlling the UAV as indicated by the original radio signals. This is used when the UAV is controlled by a pilot. The Radio Frequency (RF) transmitter is used to transmit the behaviour of the UAV to the ground station, to aid in the validation of the proposed algorithm.

Figure 2 illustrates the software architecture of the vision-based controller. The proposed system is divided into four software algorithms, namely 1) Object detection, 2) Landing, 3) stabilization, and 4) flight controller, thereof flight controller is executing on the low-performance processor and the others are executing on high-performance processor (it is distinguished by different block colors in the figure).

Landing algorithm consists of distance estimation algorithm and the landing controller part. The input of distance estimation algorithm is the location of the detected object in the image that is represented as an boundary box on the image. This function first calculates the distance between the UAV and the object, then it calculates the altitude of the UAV based on this calculated distance. Using the 2D camera gimbal the camera always points downward that makes the distance estimation algorithm possible to calculate the altitude, i.e., 'Z' of the UAV, needed for landing control. The calculated altitude, i.e., 'Z' value in Figure 2, will be subtracted from the target altitude Z_{ref} to calculate the 'Z' error to be corrected in the landing controller. The output of the landing controller is the throttle adjustment that will be passed to the flight controller unit.

Stabilization algorithm consists of object location calculator and a stabilization controller. The input of object location calculator is the location of the detected object in the captured image. Based on this location, the 'X' and 'Y' location of the UAV will be calculated. Then, these values will be subtracted from the reference 'X_{ref}' and 'Y_{ref}' preparing the error to be corrected in stabilization control algorithm. The output of the stabilization control algorithm are the roll/pitch angles that will be passed to the flight controller.

In the following each control algorithm will be discussed in more details.

IV. PROPOSED CONTROL ALGORITHMS

A. OBJECT DETECTION ALGORITHM

After receiving the captured image from the camera, a convolutional neural network (CNN) algorithm is applied for object detection. In this work, a MobileNet object detector is used that is a modified version of Single Shot Multibox (SSD) object detection algorithm [38].

MobileNet has been developed by [39] to work on mobile and embedded platforms used for vision processing. The MobileNet is based on a depthwise separable convolution. Basically, MobileNet is a class of lightweight deep

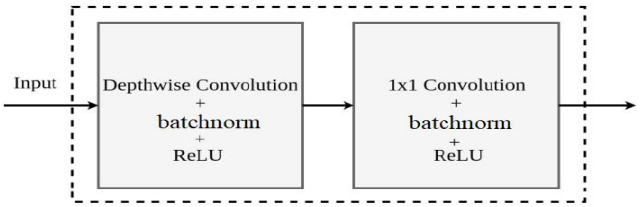


FIGURE 3. MobileNet architecture.

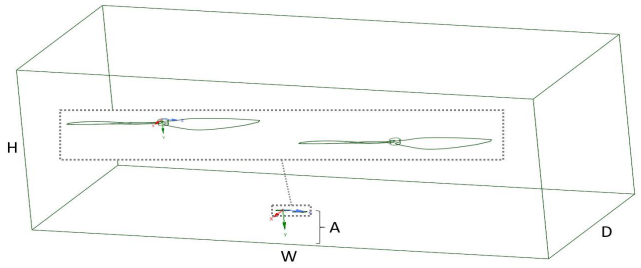


FIGURE 4. 3D geometry for CFD simulations.

convolutional neural networks that are vastly smaller in size and faster in performance than many other popular models. Depth-wise separate convolutions first apply a single filter on each input to filter the input data, followed by 1×1 convolutions which combine these filters into a set of output features. These depth-wise separable layers almost mimic the function of typical convolution layers but with much faster speed and with a slight difference (typical convolution filters and combines into output features both, but in depth-wise separable convolution this is divided into two layers, one separate layer for filtering and one separate layer for combining). This minimizes the model size and reduces computational power demands. All layers are followed by a batchnorm and ReLU nonlinearity except the final fully connected layer which feeds into a softmax layer for classification having no nonlinearity. In this paper we used 28 neural network layers without Counting depth-wise and point-wise convolutions. Figure 3 demonstrates the general architecture of MobileNet.

B. LANDING ALGORITHM

1) PREDICTION OF THE GROUND EFFECT BY CFD

At low altitudes, the downwash of air from the UAV rotors hits the ground and slows down prior to turning sideways along the ground profile. According to Bernoulli's law, this causes an increase in static pressure between the UAV frame and the ground surface, which then contributes to the UAV force balance as an additional lift term in near-ground conditions.

For multicopter UAVs, the strength of the ground effect not only depends on the altitude, but also on the propellers' speeds of rotation, their size and shape, and their relative placement: The flow fields induced by the rotors are intertwined. Analytical models exist for single rotors [40], and data-based empirical models have also been devised for multi-rotor control [41]. In the present work, CFD simulations were employed for predicting the upward lift force at different altitudes, and tuning the controller parameters.

TABLE 1. CFD model conditions.

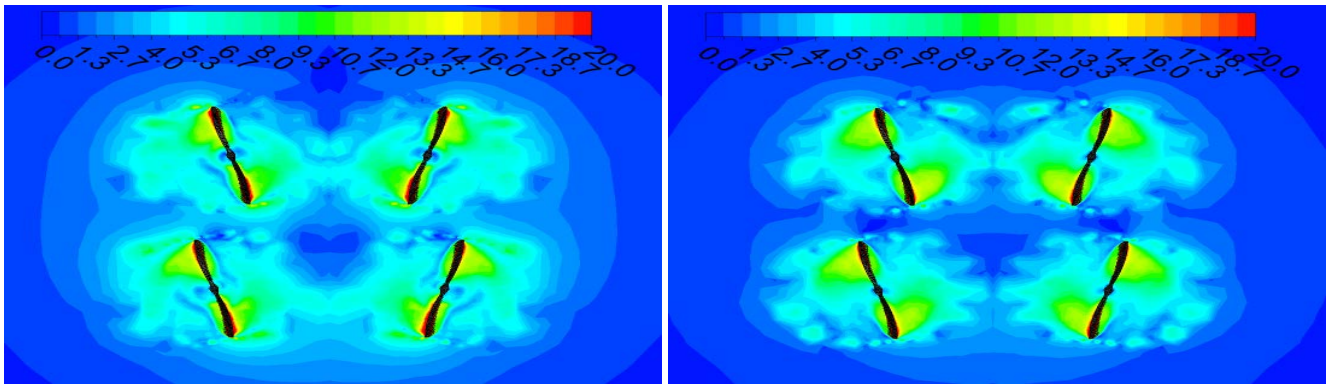
Zone	Boundary or cell zone condition
Top	Pressure inlet (0 Pa total)
Sides	Pressure outlet (0 Pa total, backflow)
Bottom	No-slip wall
Symmetry	Symmetry
Propellers	Moving reference frame (given f_{rot})
Blades	Moving no-slip wall (given f_{rot})

CFD modeling was used for predicting the strength of the ground effect at different UAV flight altitudes A [mm] and propeller rotation speeds f_{rot} [rpm]. Figure 4 shows the 3D geometry model used in the CFD simulations. In the rectangular computational domain, $W = 15267$ mm, $D = 6000$ mm and H varies according to the rotors' midpoint altitude $A = 50$, 100, 250, 1000 or 1500 mm, such that $H - A = 4230$ mm. Such a large fluid domain is necessary for ensuring that the artificial boundaries do not affect the solution near the rotors. However, to keep the model computationally sufficiently simple, the UAV frame was excluded from the model and only two of the four rotors were directly modeled; The remaining two are resolved through a symmetry plane (the front face in Figure 4). The distance from the two rotors to the symmetry plane is 272.5 mm and 232.5 mm, respectively, and they are 514.1 mm apart from each other. The rotor blades were modeled in ANSYS SpaceClaim using measurements from the UAV.

ANSYS Fluent 2019R3 was used for both creating the spatial discretization (mesh) and for solving the Reynolds-Averaged Navier-Stokes equations in the steady state. The polyhedral mesh used in the solution consists of roughly 115000 volume cells, with an appropriate boundary layer structure on the walls (cf. Figure 6). Air was treated as incompressible turbulent gas, using the $k-\omega$ SST turbulence specification. The boundary and cell zone conditions used in the CFD simulations are listed in Table 1.

The CFD simulation results consist of the air velocity components and static pressure values at all points in the domain. Figure 5 shows a comparison of the predicted flow velocity magnitudes at propeller midsection planes for $A = 50$ mm and $A = 1500$ mm. Here, $f_{rot} = 4000$ rpm, which is the maximum rated rotation speed for the propellers. Clearly the flow fields induced by the rotors are intertwined in near-ground conditions, whereas the rotors operate more independently away from the ground.

The upward lift on the propellers, as predicted by CFD, consists of pressure and viscous components. As expected, the CFD results show that the maximum lift is seen in near-ground conditions ($A = 50$ mm) and at the highest rotation speed ($f_{rot} = 4000$ rpm). Figure 7 displays the predicted lift in relation to that maximum force, for several altitudes and rotation speeds. It is clear that the ground effect prevails



(a) $A = 50$ mm, $f_{rot} = 4000$ rpm (b) $A = 1500$ mm, $f_{rot} = 4000$ rpm

FIGURE 5. Ground effect prediction by CFD simulations: Air velocity magnitude [$\frac{m}{s}$] at the propeller plane.

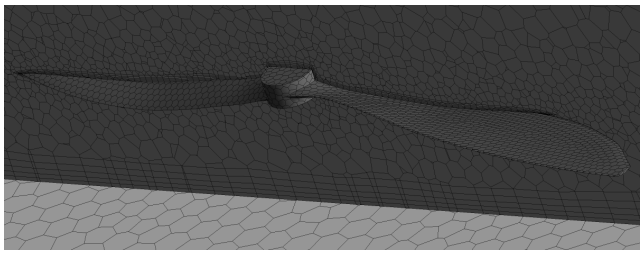


FIGURE 6. CFD mesh near a propeller.

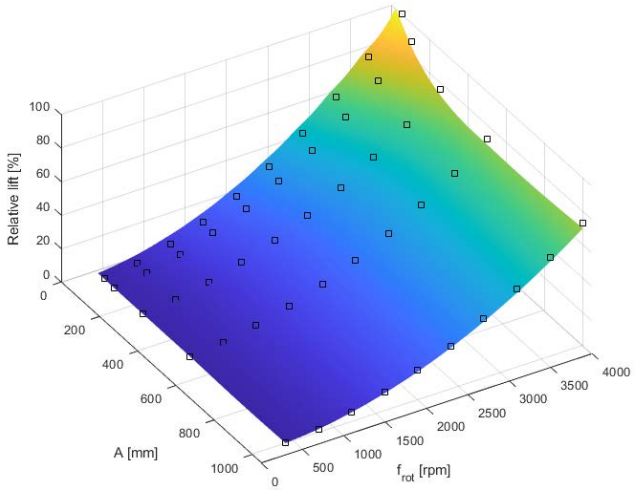


FIGURE 7. The relative lift at different altitudes and propeller speeds. The black squares denote CFD simulation results.

approximately for $A = 0 \ldots 500$ mm. At higher altitudes, the lift is approximately 70% of the observed maximum value.

2) DISTANCE ESTIMATION

The pinhole camera model [42] describes the mathematical relation between the coordinates of a point in three-dimensional space, its projection onto the image plane of an ideal pinhole camera and the focal length of the camera. Equation 1 and 2 describe the mathematical relation of the pinhole camera.

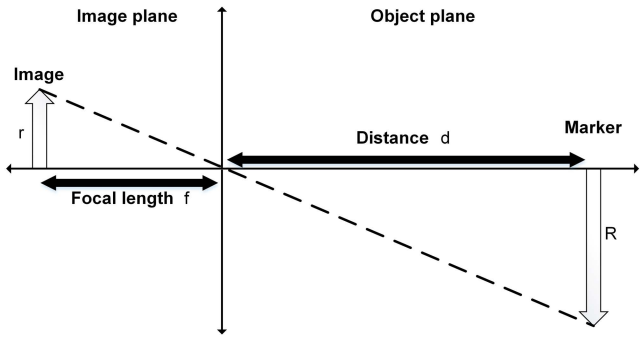


FIGURE 8. The geometry of a pinhole camera.

$$\frac{f}{d} = \frac{r}{R} \tag{1}$$

$$d = f \times \frac{R}{r} \tag{2}$$

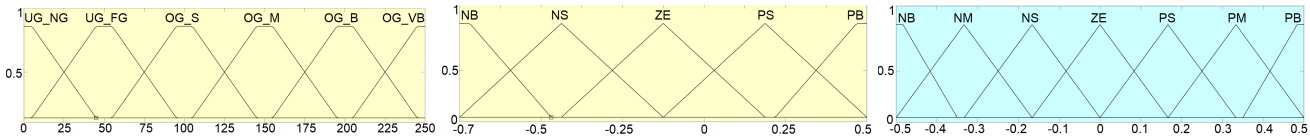
where f is the focal length of the camera, r radius of the marker in the image plane, R radius of the marker in the object plane, and d distance from camera to the object in (cm). From the previous equations, d is calculated by using trigonometry as demonstrated in Figure 8.

3) LANDING CONTROL ALGORITHM

After calculating the error between the Z and Z_{ref}, the error value is fed to the landing control as shown in Figure 2 to give out the suitable throttle adjustment value that is required to land the UAV safely on the detected object.

To account for the *ground effect* that exists near the ground while the UAV is descending, FLC is utilized to control the UAV landing. In the present work, the UAV lifting force is assumed to be divided into 1) UG (Under ground effect), and 2) OG (Outside ground effect) conditions. Here, UG is a condition where, due to the ground effect, less power is required, and OG refers to conditions other than UG.

The definition of a fuzzy set permits one to assign values to fuzzy variables as inputs and outputs. The input/output



(a) Fuzzy input (Distance(cm)) (b) Fuzzy input (Speed (m/s)) (c) Fuzzy output (Throttle Adjustment (%))

FIGURE 9. Fuzzy logic input/output membership function.

membership functions are shown in Figure 9 and the rule base is given in Table 2. A Mamdani fuzzy inference system [10] with triangular membership functions is used to apply our set of *linguistic control rules* obtained from experiments and CFD simulations. This is done by formulating the mapping from the given inputs, i.e., distance and vertical speed, to the output, throttle adjustment. In Figure 9(a), linguistic variables indicate the position of the UAV, where the fuzzy linguistic variables are defined as UG_NG (UG range Near Ground), UG_FG (UG range Far Ground), OG_S (OG range Small), OG_M (OG range Medium), OG_B (OG range Big) and OG_VB (OG range Very Big) for the corresponding distance input. The fuzzy membership functions UG-NG, UG-FG, OG-S have been constructed based on CFD predictions in Subsection IV-B1. Figure 9(b) specifies the speed of the UAV and its direction, where the linguistic variables are NB (Negative Big), NS (Negative Small), Normal, PS (Positive Small), PB(Positive Big). Figure 9(c), in turn, shows the throttle adjustment that needs to be added or subtracted from the actual throttle percentage in the flight controller as given in Equation 3, where $Trottle_{input}$ is the throttle value of flight controller, $Trottle_{pre}$ is the previous throttle input, and $Trottle_{adj}$ is the throttle adjustments values from landing controller. The linguistic variables of the output are NB, NM (Negative Medium), NS, Normal, PS, PM (Positive Medium), and PB.

To tune a FLC, the ranges of its parameters can be either adjusted manually or it can be multiplied by a gain value as given in Equation 4, where the throttle adjustment $Throttle_{adj}$ is the calculated output of the FLC as given, K is the tuning gain, i is the input number, μ_i is the corresponding membership value, μ is the fuzzy input, and n is the total number of inputs. The FLC uses *MIN* for *t-norm* operation, *Max* for *s-norm* operation, *MAX* for aggregation, *MIN* for implication, and *Centroid* for defuzzification.

$$Throttle_{input} = Throttle_{pre} \pm Throttle_{adj} \tag{3}$$

$$Throttle_{adj} = K\left(\frac{\sum_{i=1}^{n} \mu_i \mu(i)}{\sum_{i=1}^{n} \mu(i)}\right) \tag{4}$$

4) LANDING RULE BASE EXPLANATION

The FLC rule base is designed to continuously monitor the inputs; UAV's speed and its distance, and the UG range during landing process, to provide an efficient and quick landing. The control algorithms of the FLC which is used to run during the landing process is as follow:

TABLE 2. Fuzzy logic rule base for safe landing.

Distance /Speed	Mode B		Mode A			
	UG_NG	UG_FG	OG_S	OG_M	OG_B	OG_VB
NB	PB	PB	PM	PM	PM	PM
NS	PB	PM	PS	PS	PS	PS
ZE	ZE	ZE	ZE	ZE	ZE	ZE
PS	NB	NM	NS	NS	NS	NS
PB	NB	NB	NM	NM	NM	NM

- If the UAV is in the OG range, a normal landing algorithm will be applied to maintain a fixed speed while landing (−0.125m/s).
- When the UAV reaches the UG region, a special speed control algorithm based on the position of the UAV will be triggered to overcome the effect of the ground effect while landing.
- The FLC checks the moving direction of the UAV, in which it will send a high negative throttle adjustment values if it was moving upwards in order to change its direction.
- If landing speed increases or decreases, the FLC will use the throttle adjustment to make it constant.

The rule base shown in Table 2, works in two different modes based on the UAV distance, **Mode A** and **Mode B**.

Mode A is the normal operation mode of the UAV when it is outside the UG region, i.e., the movement is mainly horizontal and the view shouldn't be focused in one specific object. Moreover in this mode there is not any effect of the ground, as an external environmental effect, on the controller system. In this mode, the throttle adjustment values work between NS to PS to keep a constant landing speed of −0.125m/s, until any sudden change in speed or direction of the UAV occurs where it will generate a throttle adjustment corresponding to this change (NM to PM).

Mode B is the second scenario where the UAV is within the UG range. Through this situation, the internal and external state changes and the environmental information is mainly high, due to focus on specific area, and the speed is low. In this region, the throttle adjustment output is shifted to higher gains values according to the UAV position whether it's very close to the ground (UG_NG) or far from ground (UG_FG). In the UG_FG, the throttle adjustment is shifted from (NS-PS) to (NM-PM) to overcome the ground effect and constrain the landing speed of -0.125m/s. In the UG_NG, the ground effect gets more effective which will lead to the decrease of landing velocity until the UAV keeps hovering

above the ground without being able to land. To overcome this problem, the throttle adjustment is shifted again to higher values (NB-PB) to be able to land safely. The transition region between **Mode A** and **Mode B** needs to be smooth to avoid any kind of abrupt speed change. This transition response depends on the overlap region between UG_FG and OG_S fuzzy linguistic variables.

C. STABILIZATION ALGORITHM

1) OBJECT LOCATION CALCULATOR

The output of the object detection algorithm is represented as an boundary box on the image. From this box, the current location of the object x, y and its *width* and *height* are extracted. The location of the detected object is represented in pixel coordinates (x, y). These coordinates are changed so that the center coordinates are converted from $(\frac{width}{2}, \frac{height}{2})$ to $(0, 0)$ by using Equation 5 and Equation 6, where X and Y are the new position in pixels, *width* and *height* are the resolution of the input image [43]. These values are fed to the stabilization control algorithm.

$$X = x - \frac{width}{2} \tag{5}$$

$$Y = y - \frac{height}{2} \tag{6}$$

2) STABILIZATION CONTROL ALGORITHM

In Figure 2, once the object is detected, its position is given to the object stabilization block to start the stabilization process. In this work, a Fuzzy-PID controller is utilized to overcome the instability of the UAV, especially under environmental disturbances such as wind. Two inputs are given to the Fuzzy-PID, position (*pos*) and change of position ($\triangle pos$), and three outputs are acquired, K_{pf}, K_{if} and K_{df} which are determined by a set of fuzzy rules that are used to adapt the K_p, K_i and K_d gains of the PID controller as in Equation 7-9, where G is a gain value that can be used to for tuning the output of the Fuzzy-PID, P is the sum of the proportional gain K_p and the K_{pf} gain, I is the sum of the integral gain K_i and the K_{if} gain, and D is the sum of the derivative gain K_d and the K_{df} gain. The output of the Fuzzy-PID is described in Equation 10, where the error $e(t) = pos$ and $y(t)$ is the output of the system, which in this system represents the Roll/Pitch angles of the UAV. Figure 10 depicts the basic structure of the Fuzzy-PID controller.

$$P = (G \times K_{pf}) + K_p \tag{7}$$

$$I = (G \times K_{if}) + K_i \tag{8}$$

$$D = (G \times K_{df}) + K_d \tag{9}$$

$$y(t) = (P \times e(t)) + (I \times \int_0^t e(\tau)d(\tau)) + (D \times \frac{de(t)}{dt}) \tag{10}$$

Figure 11 shows the inputs and outputs membership function of the Fuzzy-PID, where Figure 11(a) indicates the position of detected object in pixels for x position and y position. Moreover, it displays the $(\triangle pos)$ which ranges

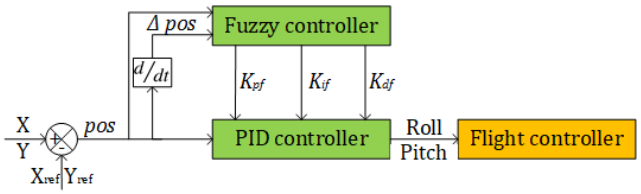
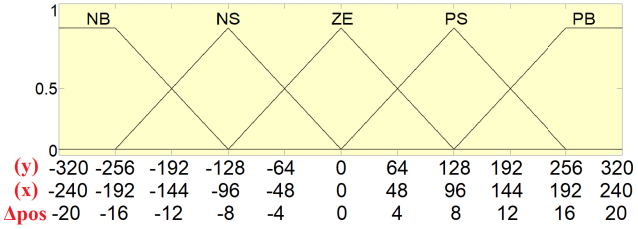
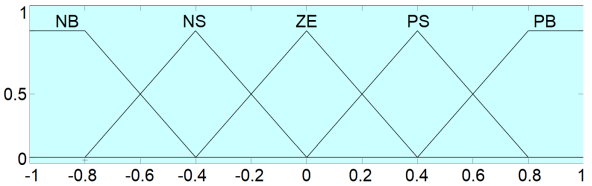


FIGURE 10. Fuzzy-PID basic structure.



(a) Fuzzy input $(pos(y, x), \triangle pos)$



(b) Fuzzy output (K_{pf}, K_{if} and K_{df})

FIGURE 11. Fuzzy-PID input/output membership function.

from -20 to 20. The fuzzy linguistic variables of the inputs are defined as NB, NS, ZE (Zero), PS, and PB. Figure 11(b) specifies the output of Fuzzy-PID (K_{pf}, K_{if} and K_{df}) that is added or subtracted to the fixed gains K_p, K_i and K_d, as illustrated in Equations 7-9, and its linguistic variables are defined as NB, NS, ZE (Zero), PS, and PB. K_p, K_i and K_d values are obtained from studying the effect of these parameters on the simulation model that is covered in Section V.B.2, and was further explained in our previous work [20]. The G value is by default '1', however for the sake of simplicity in tuning the Fuzzy-PID without the need to modify its membership functions, the G value can be modified to tune different systems.

3) STABILIZATION RULE BASE EXPLANATION

Designing the rule-base for the Fuzzy-PID is the most crucial part because the whole function of this controller depends on it. For this, several experiments were conducted in addition to experience to design it. Table 3-5 show the rule base that is used to utilize the P, I and D gains of the PID controller for the UAV stabilization.

The principles of constructing the Fuzzy-PID rule base are as follow:

- When *pos* is NB/PB (UAV is far from the center of the detected object) and the UAV is moving away from the detected object (e.g., *pos* is NB and $\triangle pos$ is also NB), K_{pf} should be big to increase the P gain of the PID to push the UAV towards the center of the detected object.

On the other hand, K_{df} and K_{if} ought to be NB, so that I and D become very small for better following performance and to avoid any oscillation.

- When pos is NB/PB and the UAV moves towards the center of the detected object (e.g., pos is NB and $\triangle pos$ is PS/PB), K_{pf} is slightly increased when $\triangle pos$ is PS to slightly increase its speed. On contrast, if $\triangle pos$ is PB, which means it moves fast towards the center of the detected object, a small negative value (NS) is given to slightly decrease the P gain in order to reduce the UAV speed to be able to keep up with the center of the detected object. In both cases, K_{if} is either ZE or NS to avoid any integral saturation or overshoot.
- When pos is NS/PS (UAV is close to the center of the detected object) and $\triangle pos$ is NB/PB, this indicates that the UAV is moving in high speed away from the object. For this, K_{pf}, K_{if} and K_{df} are treated as the first principle to avoid losing the detected object.
- When pos is NS/PS, and UAV moves towards the detected object, P gain is decreased by a small value and D gain is slightly increased for steady.
- When pos is ZE, only K_{if} and K_{df} will affect the PID controller and the values of I and D gains are increased to keep a minimum speed to stabilize the UAV above the center of the detected object.

TABLE 3. Fuzzy-PID rule base for K_{pf} gain.

$pos/\triangle pos$	NB	NS	ZE	PS	PB
NB	PB	PB	ZE	NB	NS
NS	PB	PS	ZE	NS	PS
ZE	PS	ZE	ZE	ZE	PS
PS	PS	NS	ZE	PS	PB
PB	NS	NB	ZE	PB	PB

TABLE 4. Fuzzy-PID rule base for K_{if} gain.

$pos/\triangle pos$	NB	NS	ZE	PS	PB
NB	NB	NB	PS	ZE	ZE
NS	NB	NS	PS	ZE	NS
ZE	NS	NS	ZE	NS	NS
PS	NS	ZE	PS	NS	NB
PB	ZE	ZE	PS	NB	NB

TABLE 5. Fuzzy-PID rule base for K_{df} gain.

$pos/\triangle pos$	NB	NS	ZE	PS	PB
NB	NB	NS	PB	PB	PS
NS	NS	ZE	PS	PS	ZE
ZE	ZE	ZE	ZE	ZE	ZE
PS	ZE	PS	PS	ZE	NS
PB	PS	PB	PB	NS	NB

V. EXPERIMENTAL SETUP

A. UAV PHYSICAL SYSTEM

In this work, a UAV multicopter system is demonstrated as shown in Figure 12. The multicopter consists of a high-performance CPU, Jetson TX2, installed on a J120 carrier board, powered with a 3S 11.1V LiPo battery that weights 120g and is connected to an ultra HD Logitech webcam through USB. The camera is attached to a 2D-gimbal that is used to make toe camera face the ground while the multicopter is landing. The proposed system also consist of a low-performance CPU, Pixhawk 2.4.8 flight controller, that is based on a 180 MHz ARM® Cortex® M4, and is equipped with a 10DoF IMU, eight PWM outputs, four UART and other different connections. The flight controller is build on an open source program "ArduPilot", which is codded in C++. Furthermore, it can be easily monitored or configured from a ground control station using an RF transmitter/receiver. The low-performance CPU is connected to the high-performance CPU using a UART communication.

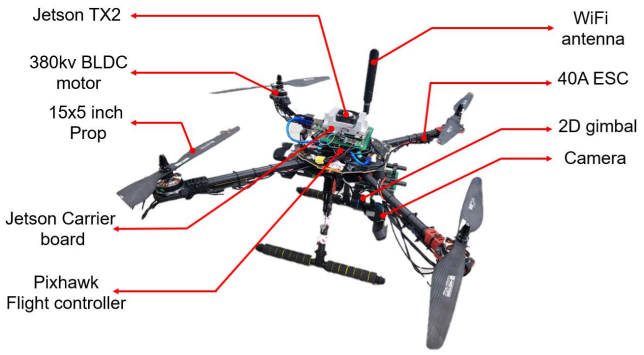


FIGURE 12. UAV multicopter practical system.

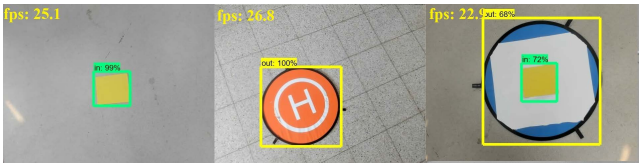
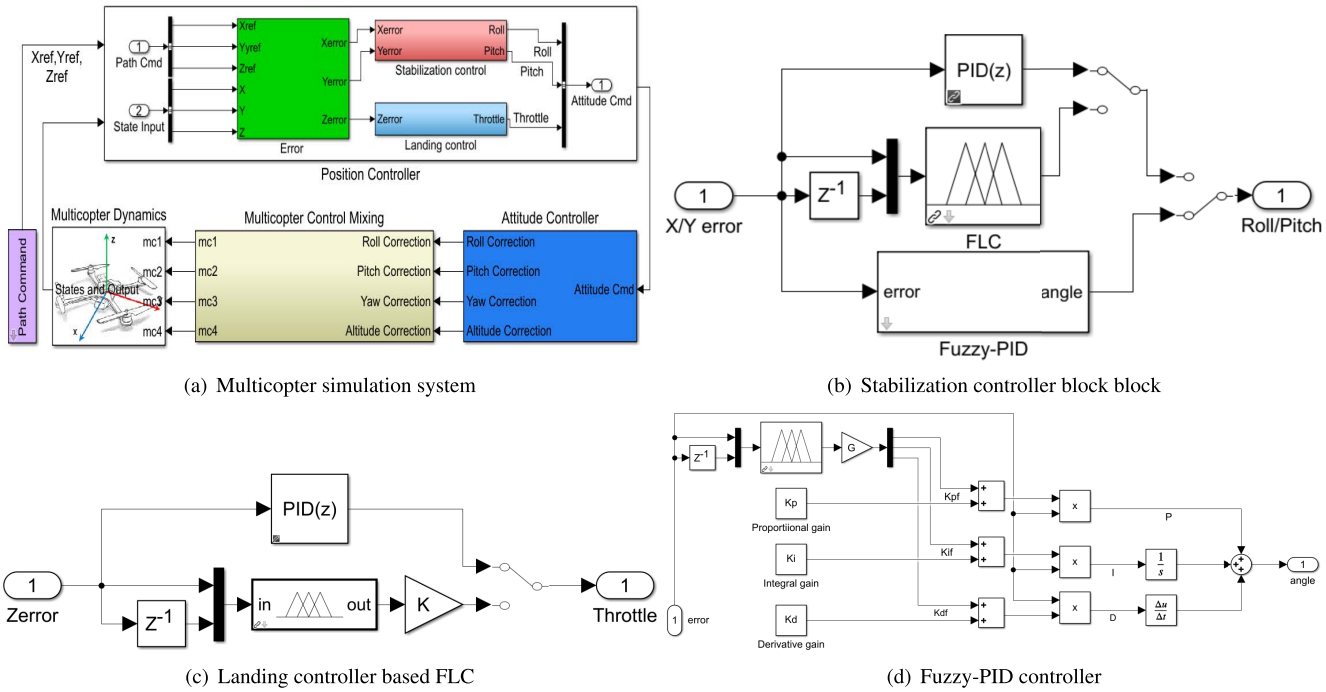


FIGURE 13. Output of the object detection algorithm.

B. OBJECT DETECTION

The CNN was trained to detect two classes, a rectangle object "in" and a circle object "out". A custom image dataset that contains 10000 images of the required images on different backgrounds has been used for training. The images have been captured by the camera installed on the multicopter and have been manually labeled. The images are divided into two parts, training and test dataset. 20% of the images are used to test the network, while the other 80% are used to train the dataset. The network training parameters values are based on [39], and is trained for 100,000 iteration until the error does not significantly decrease, with a learning rate of 0.004 and decay factor 0.95. The root mean square propagation (RMSprop) optimization algorithm is used for optimizing the loss functions. The training is conducted on a desktop with the following specification: AMD Ryzen 7 2700X 3.70 GHz CPU, 16 GB RAM, and a NVIDIA RTX



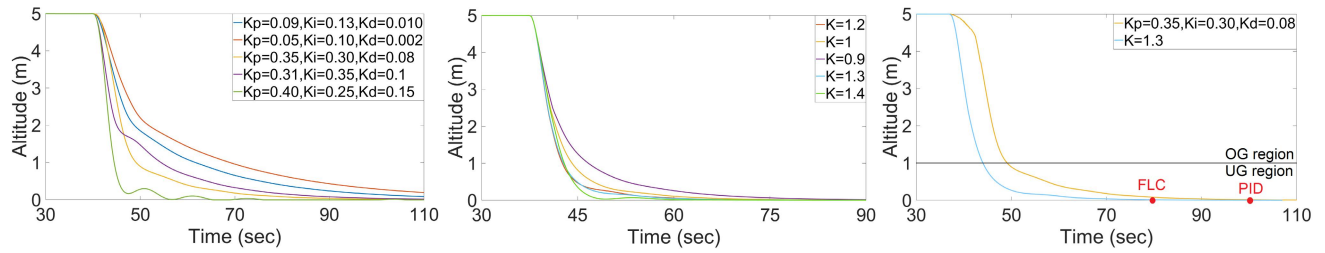
(a) Multicopter simulation system

(b) Stabilization controller block block

(c) Landing controller based FLC

(d) Fuzzy-PID controller

FIGURE 14. Simulink model of the Multicopter simulation platform.



(a) PID controller output

(b) FLC output

(c) Response Comparison of PID and FLC

FIGURE 15. Performance comparison of the FLC and PID during landing process.

2080TI GPU. The trained CNN achieved an accuracy of 84% with average fps of 25.1.

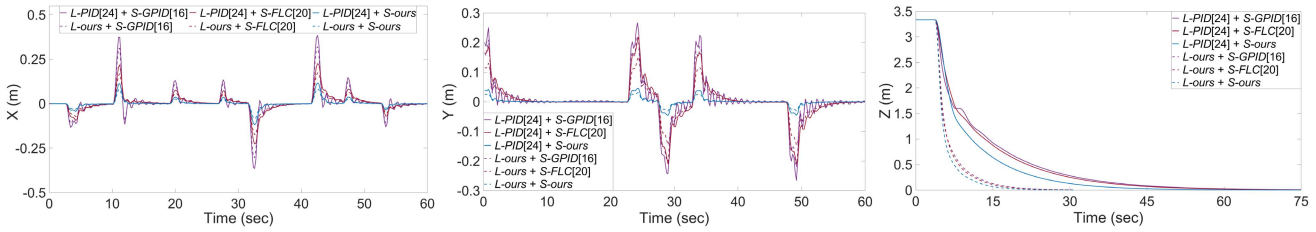
The CNN algorithm used for detecting the objects is a SSD_mobilenet_v1 which is optimized to run on embedded platforms in real-time. The SSD_mobilenet_v1 is implemented on the Jetson TX2 and coded using python programming language. The object detector is trained to detect two objects, a circle object and a rectangle object as shown in Figure 13. As illustrated in this Figure, the average fps achieved by SSD_mobilenet_v1 is 25fps which make it suitable for real-time applications.

To perform object detection and landing, the USB camera keeps on taking continuous snapshots on 640 × 480 pixels. Once the objects are detected and distance between the multicopter and the objects is estimated, landing and stabilization processes are triggered to work simultaneously to ensure the a smooth and safe landing of the multicopter over the center of the detected object.

VI. RESULTS

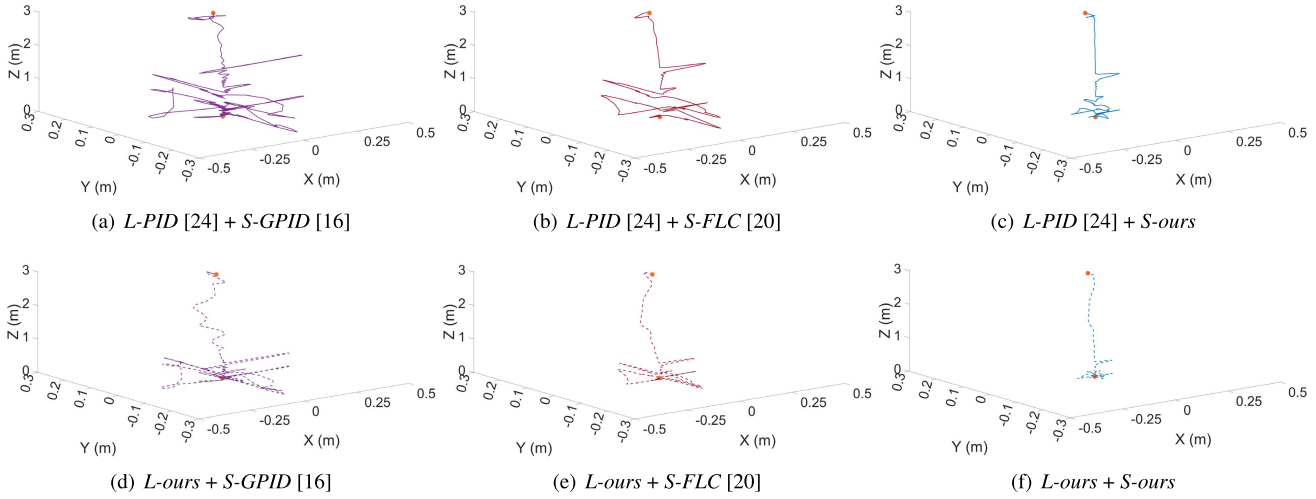
A. SIMULATION RESULTS

For evaluating the proposed technique, MATLAB/Simulink simulation environment has been used [44]. Figure 14 shows the Simulink models of the multicopter platform. Within this simulation, multicopter parameters, e.g., initial location, weight, size, etc., can be changed. The location error is calculated by taking the difference between the multicopter current position (X, Y, and Z) and the reference position (X_{ref}, Y_{ref}, and Z_{ref}). Afterwards, the error is passed to the proposed stabilization and landing control blocks thereafter the attitude commands (Roll/Pitch/Throttle) will be passed to the attitude controller block and the multicopter control mixing block, which are responsible to convert the attitude commands to actuation signals to the motors. The multicopter dynamics consists of mathematical equations about the dynamics of different parts of the multicopter. More details about these equations can be found in [45].



(a) X-axis value (b) Y-axis value (c) Z-axis value

FIGURE 16. **(X, Y, Z) location of the multicopter during the simulation period.**



(a) *L-PID* [24] + *S-GPID* [16] (b) *L-PID* [24] + *S-FLC* [20] (c) *L-PID* [24] + *S-ours*

(d) *L-ours* + *S-GPID* [16] (e) *L-ours* + *S-FLC* [20] (f) *L-ours* + *S-ours*

FIGURE 17. **Location of the multicopter in 3D spatial domain during the simulation period.**

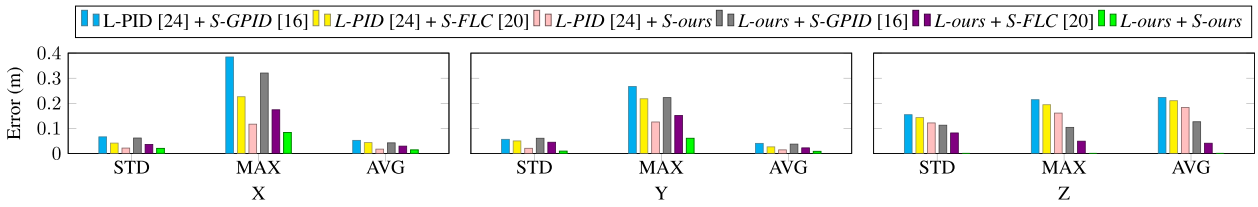


FIGURE 18. **Statistical analysis of (X, Y, Z) location of the multicopter during the simulation period.**

1) FUZZY LOGIC BASED LANDING CONTROL

To show the performance of the proposed FLC controller, a PID controller is first implemented and tested to obtain the proper gain parameters. Afterward, FLC is implemented and tested for the same Simulink model as shown in Figure 14(c).

Figure 15(a) shows the response of PID with different tuning parameters. From this figure, the landing speed changes by different PID gains. The "green line" in this figure has the fastest landing speed, however, it suffers from oscillation when reaching the ground, which will lead to the damage of the multicopter. Other responses are much safer but they suffer from lower landing speed especially near the ground due to the ground effect.

As mentioned in Section IV.B.3, a gain value K can be multiplied by the FLC output for better tuning. Figure 15(b) shows the output of FLC with different gains. The higher

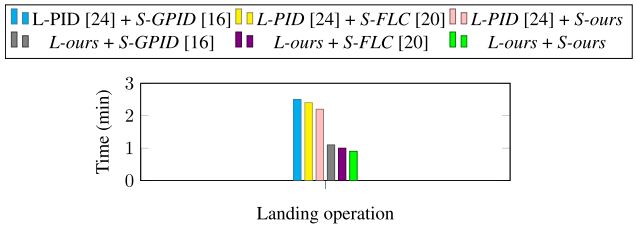


FIGURE 19. **Controllers landing time during simulation phase.**

the K, the faster the landing speed. However, when K is very high, it will cause bumping leading to the damage of the multicopter. Therefore, a proper K should be selected.

A landing comparison between the proper selected parameters of PID and FLC is shown in Figure 15(c). The total time for landing using the PID controller is 60 sec, while FLC

TABLE 6. Average error of the designed techniques during simulation period.

Error(m)/Technique	L-PID [24] + S-GPID [16]	L-PID [24] + S-FLC [20]	L-PID [24] + S-ours	L-ours + S-GPID [16]	L-ours + S-FLC [20]	L-ours + S-ours
STD	0.091	0.076	0.053	0.078	0.047	0.014
AVG	0.105	0.093	0.071	0.068	0.031	0.022
MAX	0.288	0.213	0.133	0.215	0.125	0.075

takes 40 sec. Due to the ground effect, the PID has slower response while approaching the ground (UG range) due to its fixed gains. On contrast, the FLC shows better response when landing due to its rule base which combines both speed and distance, that will ensure a smooth and safe landing process.

2) STABILIZATION AND LANDING PROCESS

In our scenario, the multicopter starts to ascend up to 3 meters above the ground surface and will hold its position for 15 seconds. After that it decides to land on a pre-specified target that has been defined on the ground by the object. We inserted random disturbance in different X, Y, and Z directions to model an uncertain environment for the multicopter.

To show the efficiency of the proposed technique with the state-of-the-art, we implemented different control algorithms for both stabilization and landing parts. For the landing part we implemented the pure PID control algorithm for UAVs in [24] and [25], called as *L-PID* in this paper, and for the stabilization algorithm we implemented a Gain-scheduled PID proposed in [16] and a fuzzy control proposed in [20] (called *S-GPID* and *S-FLC* respectively). Our proposed control algorithm in the comparison is shown by *S-ours* and *L-ours* for stabilization and landing respectively. This comparison setup forms 6 different configurations for the UAV control techniques based on two different landing controls, i.e., L-PID and L-Ours, and three stabilization control techniques, i.e., S-GPID, S-FLC, and S-ours. We discuss the results obtained from these 6 configurations in the rest of the results section.

Figures 16(a-c) show the X, Y, and Z positions of the multicopter while using different control techniques respectively. It can be seen that *S-ours* achieved better stability and settling time compared to other techniques in X-Y plane. Figure 18 shows the corresponding statistical values about the occurred error in X, Y, and Z axes, i.e., the standard deviation (STD), average/mean error (AVG), maximum error (MAX), which is responsible for returning the maximum error in our dataset, and Table 6 shows the average statistical values of the designed techniques during the landing and stabilization process. Equation 11 and Equation 12 show the mathematical formulas for calculating the corresponding statistical values shown in Figure 18, where n_i is a value in the data set that corresponds to the position of the multicopter in X, Y, and Z, and N is the total number of data points.

$$STD = \sqrt{\frac{\sum |n_i - AVG|^2}{N}} \tag{11}$$

$$AVG = \frac{\sum n_i}{N} \tag{12}$$

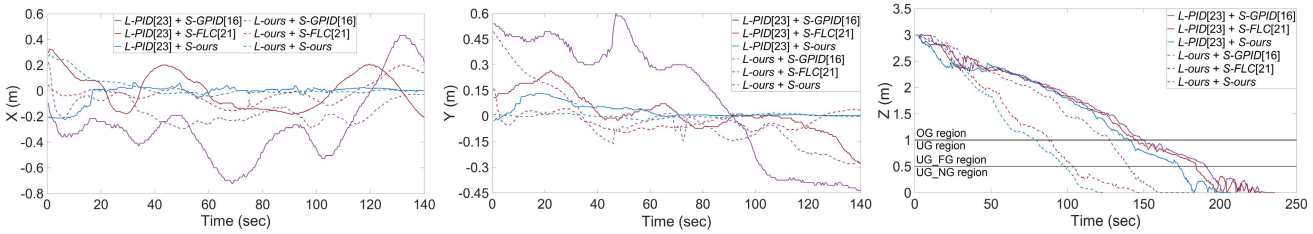
The average error for X and Y are approximately zero for all the control configurations. This is because of the random disturbances that have been injected in different direction on X-Y plane. However, the MAX and STD values show that the proposed technique can significantly reduce the disturbance error in X and Y dimensions. This can be noticed for both configurations with our proposed stabilization, i.e., *L-PID* + *S-ours* and *L-ours* + *S-ours*. In Z dimension, the landing algorithm shows its robustness by reducing the MAX and STD values of the error comparing to the other techniques.

It can be seen that the stabilization part improves the MAX and STD of error while working with the proposed landing controls, i.e., *L-ours* + *S-ours*, comparing to *L-PID* + *S-ours* and *L-ours* + *S-GPID*, and *L-ours* + *S-FLC*. This shows that the proposed landing and stabilization algorithm give the best outcome when *both techniques are working together* as shown in Figure 17. This is due to the fact that the command signals to the propellers are the ensemble of both controller commands that are merged and tuned together to manipulate the mechanical parts of the multicopter, such as propellers and other mechanical parts. Any change or disturbance of the multicopter location in any axes affects its location in other axes since the multicopter control mixing unit converts the required attitude corrections into throttle percentages for each motor. The higher the attitude corrections values, the more throttle given to each motor, which in return will lead to more disturbances of the multicopter during landing.

Figure 19 shows the landing time of the 6 different controllers. Using the full-fledged approach, i.e., *L-ours* + *S-ours*, will reduces the operation time by 6%, 8%, 51%, 59%, and 65% comparing to (*L-ours* + *S-FLC*, *L-ours* + *S-GPID*, *L-PID* + *S-ours*, *L-PID* + *S-FLC*, and *L-PID* + *S-GPID*) respectively. The reason is that by reducing the amount of disturbances and providing more stability, the multicopter can reach the target in a faster and smoother way comparing to the other techniques.

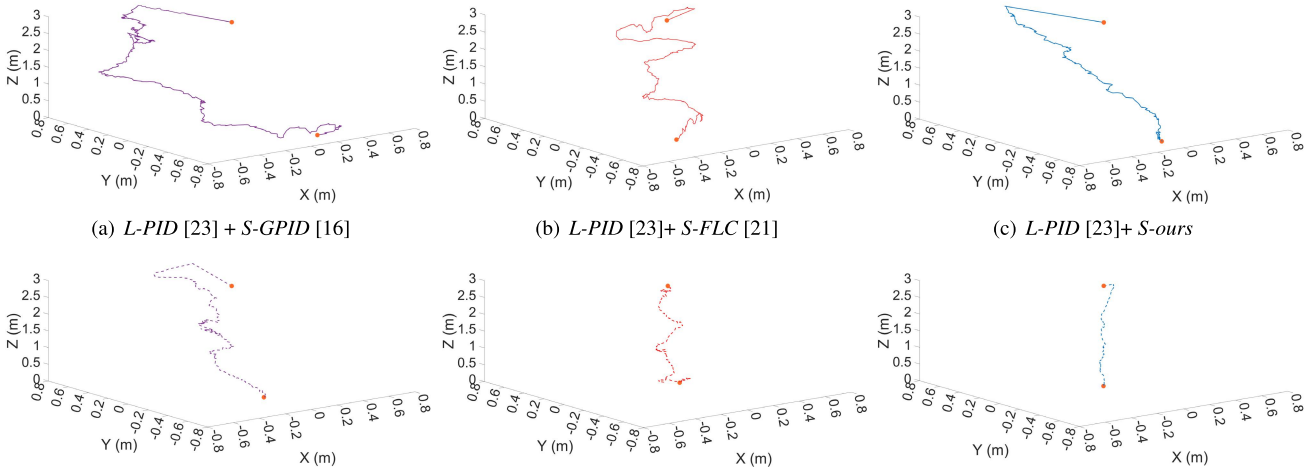
B. RESULTS ON THE REAL PLATFORM

Figures 20(a-c) show the X, Y, and Z positions of the multicopter while using different control techniques in our experiment on a real environment. Figure 22 shows the corresponding statistical values about the occurred error in X, Y, and Z axes, i.e., the standard deviation (STD), maximum error (MAX), and average error (AVG), while Table 7 shows the average statistical values of the designed techniques during the landing and stabilization process throughout the experimental period. As it can be seen, opposite to the simulation results the average disturbance in X and Y direction is not



(a) Stabilization comparison in X-axis (b) Stabilization comparison in Y-axis (c) Landing comparison

FIGURE 20. **(*X*, *Y*, *Z*) location of the multicopter during the experimental period.**



(a) *L-PID* [23] + *S-GPID* [16] (b) *L-PID* [23]+ *S-FLC* [21] (c) *L-PID* [23]+ *S-ours*

(d) *L-ours* + *S-GPID* [16] (e) *L-ours* + *S-FLC* [21] (f) *L-ours* + *S-ours*

FIGURE 21. **Location of the multicopter in 3D spatial domain during the experimental period.**

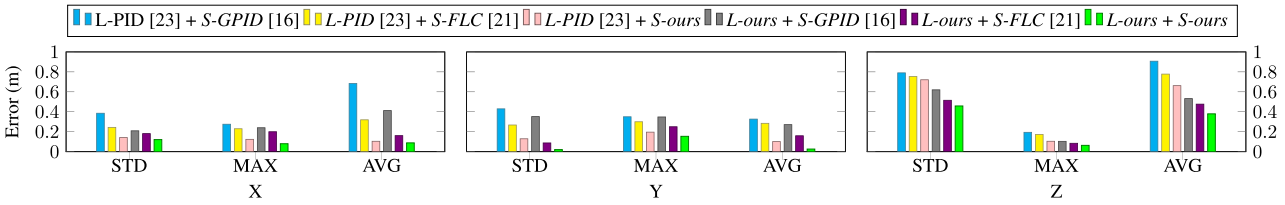


FIGURE 22. **Statistical analysis of (*X*, *Y*, *Z*) location of the multicopter during the experimental period.**

TABLE 7. **Average error of the designed techniques during experimental phase.**

Error(m)/Technique	L-PID [23] + S-GPID [16]	L-PID [23] + S-FLC [21]	L-PID [23] + S-ours	L-ours + S-GPID [16]	L-ours + S-FLC [21]	L-ours + S-ours
STD	0.538	0.422	0.331	0.324	0.265	0.197
AVG	0.641	0.463	0.289	0.405	0.214	0.164
MAX	0.273	0.234	0.142	0.227	0.178	0.099

zero, showing the environmental condition is not always following a uniform random pattern. Consequently, the multicopter shows much unstable behaviour with respect to environmental and control stimuli. It can be seen that the proposed technique can significantly reduce this unexpected/unwanted disturbance.

L-PID + *S-GPID* shows the poorest response among all the tested techniques. Although the *S-GPID* is easy to implement, entailing a fast way to control the system, it will fail to control the multicopter in case of high disturbances, especially when combined with *L-PID* because of its linearity which will require continuous change of the PID gains. On the other hand, *S-FLC* have better outcome compared to *S-GPID* because this controller considers both the error value and its derivative as the inputs targeting to minimize the error between the reference and the actual output. Since the proposed Fuzzy-PID controller *S-ours* auto-tunes the PID gains, according to its two inputs 1) position and 2) change of the position of the multicopter, it can rapidly reduce the error between the multicopter and center of target with

TABLE 8. Computational timing comparison of the designed techniques.

Task (msec)/Technique	L-PID [23] + S-GPID [16]	L-PID [23] + S-FLC [21]	L-PID [23] + S-ours	L-ours + S-GPID [16]	L-ours + S-FLC [21]	L-ours + S-ours
Object detection	29.3	29.1	28.8	29.4	29.1	28.4
Stabilization	2.9	3.3	4.1	3.1	3.4	4.3
landing	1.6	1.5	1.6	2.5	2.4	2.7
Total time	33.8	33.9	34.5	35.1	34.9	35.4

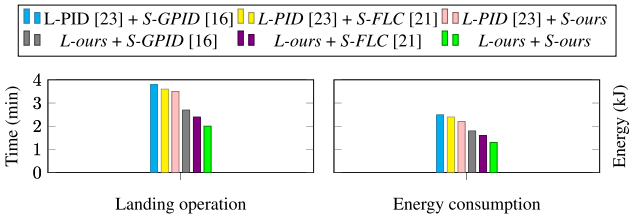


FIGURE 23. Controllers landing time and energy consumption during experimental period.

a better stability. Thus, the Fuzzy-PID controller combines the pros of both a PID controller and a FLC; resulting in the best outcome in contrast with the other techniques.

Figure 23 shows the overall landing time that is taken while using different techniques. In general the proposed technique outperforms the average error values by 20%, 45%, 18%, 51%, and 78% and the overall landing time by 11%, 19%, 38%, 43%, and 47% compared to (*L-ours + S-FLC*, *L-ours + S-GPID*, *L-PID + S-ours*, *L-PID + S-FLC*, and *L-PID + S-GPID*) respectively. One main reason to achieve such a considerable improvement for the overall landing time in comparison with the similar results over the simulation phase, shown in Figure 19, is that the high disturbance that the multicopter has been experienced in real platform is much higher than the simulation. Lower disturbance mainly provides better opportunity for the vision camera and object detection to follow the object that decrease the amount of object detection time and error in loop summing and vice versa.

Figure 23 also compares the energy consumption of the proposed technique with the other techniques. The proposed technique can reduce the overall energy consumption of the multicopter by 15%, 21%, 28%, 36%, and 42% comparing to (*L-ours + S-FLC*, *L-ours + S-GPID*, *L-PID + S-ours*, *L-PID + S-FLC*, and *L-PID + S-GPID*) respectively. The reason is that removing the disturbances can help to reduce the energy from both computational and mechanical energy consumption. Less disturbances provide more stable image that makes the image denoising and object detection faster and, consequently, more energy efficient. Also, any unwanted disturbance from both environment and control actuation, can cause a reaction from the controller with energy consumption in mechanical part. Therefor reducing the disturbance can reduce the energy consumption of the mechanical part.

In general, the FLC is a complicated controller that requires a long computation time. In this work, the Jetson TX2 is responsible for performing three different tasks, where two of them are executed simultaneously as shown in figure 2. The object detection part takes a large execution time due to its huge data size, and implementing two other algorithms, e.g., FLC and Fuzzy-PID controllers, in parallel with the object detection will cause processing delay. For this, MATLAB/Simulink is used to generate a lookup table that calculates the output of the FLC and Fuzzy-PID controllers corresponding to different input values instead of the real-time FLC. Table 8 shows the average time for executing the three algorithms; object detection, landing, and stabilization, of the designed controllers. As can be seen in Table 8, due to the usage of lookup table, the computation time to execute a FLC and a Fuzzy-PID is slightly larger than a Gain-Scheduled PID controller in the stabilization algorithm (average 0.35 msec and 2.35 msec respectively), and the difference between a FLC and a PID controller is 0.96 msec on average in the landing process. Combining both timing, in addition to the object detection algorithm and calculating the total time of each designed technique, our proposed controller $L-ours+S-ours$ execution time is slightly higher and can be neglected in comparison to other techniques. Furthermore, having a better stability, settling time, and more energy efficient compared to the other approaches, makes the proposed technique more suitable for such application.

VII. CONCLUSION

In this paper, two fuzzy control techniques for stabilization and landing are proposed that considers manipulating the multicopter's roll and pitch. The mission (landing) for the UAV is defined to track (land on) an object that is detected by a deep learning object detection algorithm. The inputs of the controller are the location and speed of the UAV that has been calculated based on the location of the detected object. Two fuzzy control techniques are proposed to control the stabilization of the UAV during the mission and to overcome the ground effect during the landing period. Several experiments are performed, and statistical analysis of the UAV location are acquired. The obtained results show that the proposed technique has low computation time, better performance, and less energy consumption for both two operational modes of mission and landing in comparison with other technique.

Future studies could improve the current work by studying the effect of using a reinforcement learning technique to auto-tune the PID controllers. Furthermore, using an adaptive FLC for landing process with a simplified rule base technique to overcome the ground effect is also considered to improve this work.

REFERENCES

[1] A. Rohan, M. Rabah, and S.-H. Kim, "Convolutional neural network-based real-time object detection and tracking for parrot AR drone 2," *IEEE Access*, vol. 7, pp. 69575–69584, 2019.

[2] B. Lee, V. Saj, M. Benedict, and D. Kalathil, "A deep reinforcement learning control strategy for vision-based ship landing of vertical flight aircraft," in *Proc. AIAA Aviation Forum*, Aug. 2021, p. 3218.

[3] A. M. Jawad, H. M. Jawad, R. Nordin, S. K. Gharghan, N. F. Abdullah, and M. J. Abu-Alshaeer, "Wireless power transfer with magnetic resonator coupling and sleep/active strategy for a drone charging station in smart agriculture," *IEEE Access*, vol. 7, pp. 139839–139851, 2019.

[4] H. Yu, F. Zhang, and P. Huang, "CSLAM and GPS based navigation for multi-UAV cooperative transportation system," in *Proc. Int. Conf. Intell. Robot. Appl.* Cham, Switzerland: Springer, 2019, pp. 315–326.

[5] Y. Lu, Z. Xue, G. S. Xia, and L. Zhang, "A survey on vision-based UAV navigation," *Geo-Spatial Inf. Sci.*, vol. 21, no. 2, pp. 21–32, 2018.

[6] L. Yang, X. Feng, J. Zhang, and X. Shu, "Multi-ray modeling of ultrasonic sensors and application for micro-UAV localization in indoor environments," *Sensors*, vol. 19, no. 8, p. 1770, Apr. 2019.

[7] J. Q. Cui, S. Lai, X. Dong, and B. M. Chen, "Autonomous navigation of UAV in foliage environment," *J. Intell. Robotic Syst.*, vol. 84, nos. 1–4, pp. 259–276, Dec. 2016.

[8] G. Balamurugan, J. Valarmathi, and V. P. S. Naidu, "Survey on UAV navigation in GPS denied environments," in *Proc. Int. Conf. Signal Process., Commun., Power Embedded Syst. (SCOPES)*, Oct. 2016, pp. 198–204.

[9] Y. Gui, P. Guo, H. Zhang, Z. Lei, X. Zhou, J. Du, and Q. Yu, "Airborne vision-based navigation method for UAV accuracy landing using infrared lamps," *J. Intell. Robot. Syst.*, vol. 72, no. 2, pp. 197–218, 2013.

[10] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *Int. J. Man-Machine Stud.*, vol. 7, no. 1, pp. 1–13, 1975.

[11] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 46–56, Sep. 2012.

[12] L. Merino, F. Cabalero, J. R. Martínez-de Dios, J. Ferruz, and A. Ollero, "A cooperative perception system for multiple UAVs: Application to automatic detection of forest fires," *J. Field Robot.*, vol. 23, nos. 3–4, pp. 165–184, 2006.

[13] P. Doherty and P. Rudol, "A UAV search and rescue scenario with human body detection and geolocalization," in *Proc. Australas. Joint Conf. Artif. Intell.* Cham, Switzerland: Springer, 2007, pp. 1–13.

[14] H. Lim and S. N. Sinha, "Monocular localization of a moving person onboard a quadrotor MAV," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2015, pp. 2182–2189.

[15] H. Huang, A. V. Savkin, and C. Huang, "Reliable path planning for drone delivery using a stochastic time-dependent public transportation network," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 4941–4950, Aug. 2021.

[16] M. Rabah, A. Rohan, M.-H. Haghbayan, J. Plosila, and S.-H. Kim, "Heterogeneous parallelization for object detection and tracking in UAVs," *IEEE Access*, vol. 8, pp. 42784–42793, 2020.

[17] M. Kamel, S. Verling, O. Elkhatib, C. Sprecher, P. Wulkop, Z. Taylor, R. Siegwart, and I. Gilitschenski, "The voliro omniorientational hexacopter: An agile and maneuverable tiltable-rotor aerial vehicle," *IEEE Robot. Autom. Mag.*, vol. 25, no. 4, pp. 34–44, Dec. 2018.

[18] (2018). *Px4-Community*. [Online]. Available: https://docs.px4.io/en/

[19] J. Ye, C. Fu, F. Lin, F. Ding, S. An, and G. Lu, "Multi-regularized correlation filter for UAV tracking and self-localization," *IEEE Trans. Ind. Electron.*, vol. 69, no. 6, pp. 6004–6014, Jun. 2022.

[20] M. Rabah, A. Rohan, Y.-H. Han, and S. H. Kim, "Design of fuzzy-PID controller for quadcopter trajectory-tracking," *Int. J. Fuzzy Logic Intell. Syst.*, vol. 18, no. 3, pp. 204–213, 2018.

[21] R. N. Jácome, H. L. Huertas, P. C. Procel, and A. G. Garcés, "Fuzzy logic for speed control in object tracking inside a restricted area using a drone," in *Developments and Advances in Defense and Security* (Smart Innovation, Systems and Technologies). Singapore: Springer, 2020, pp. 135–145.

[22] D. Lee, J. Zhou, and W. T. Lin, "Autonomous battery swapping system for quadcopter," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2015, pp. 118–124.

[23] (2020). *Ardupilot Autopilot Software Suite*. [Online]. Available: https://ardupilot.org/ardupilot

[24] C. Chen, S. Chen, G. Hu, B. Chen, P. Chen, and K. Su, "An auto-landing strategy based on pan-tilt based visual servoing for unmanned aerial vehicle in GNSS-denied environments," *Aerosp. Sci. Technol.*, vol. 116, Sep. 2021, Art. no. 106891.

[25] A. Kumar, "Real-time performance comparison of vision-based autonomous landing of quadcopter on a ground moving target," *IETE J. Res.*, pp. 1–18, Aug. 2021.

[26] A. Borowczyk, D.-T. Nguyen, A. P.-V. Nguyen, D. Q. Nguyen, D. Saussie, and J. Le Ny, "Autonomous landing of a multirotor micro air vehicle on a high velocity ground vehicle," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 10488–10494, 2017.

[27] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural lander: Stable drone landing control using learned dynamics," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 9784–9790.

[28] N. Farah, M. H. N. Talib, Z. Ibrahim, Q. Abdullah, O. Aydoğdu, M. Azri, J. B. M. Lazi, and Z. M. Isa, "Investigation of the computational burden effects of self-tuning fuzzy logic speed controller of induction motor drives with different rules sizes," *IEEE Access*, vol. 9, pp. 155443–155456, 2021.

[29] N. Farah, M. H. N. Talib, and N. S. M. Shah, "A novel self-tuning fuzzy logic controller based induction motor drive system: An experimental approach," *IEEE Access*, vol. 7, pp. 68172–68184, 2019.

[30] Q. A. Tarbosh, O. Aydogdu, N. Farah, M. H. N. Talib, A. Salh, N. Cankaya, F. A. Omar, and A. Durdu, "Review and investigation of simplified rules fuzzy logic speed controller of high performance induction motor drives," *IEEE Access*, vol. 8, pp. 49377–49394, 2020.

[31] W. Dwiono, A. J. Taufiq, and W. Winarso, "Simple implementation of fuzzy controller for low cost microcontroller," in *Proc. Int. Conf. Artif. Intell. Inf. Technol. (ICAIIT)*, Mar. 2019, pp. 26–30.

[32] V. K. Lakshminarayan and J. D. Baeder, "Computational investigation of microscale shrouded rotor aerodynamics in hover," *J. Amer. Helicopter Soc.*, vol. 56, no. 4, pp. 1–15, Oct. 2011.

[33] J. Li, G. Lei, Y. Xian, and X. Wang, "Research on ground effect of shipborne flying-wing UAV," in *Proc. 10th Int. Conf. Comput. Intell. Secur.*, Nov. 2014, pp. 685–688.

[34] S. Prothin, C. F. Escudero, N. Doué, and T. Jardin, "Aerodynamics of MAV rotors in ground and corner effect," *Int. J. Micro Air Vehicles*, vol. 11, Sep. 2019.

[35] C. Paz, E. Suárez, C. Gil, and C. Baker, "CFD analysis of the aerodynamic effects on the stability of the flight of a quadcopter UAV in the proximity of walls and ground," *J. Wind Eng. Ind. Aerodynamics*, vol. 206, Nov. 2020, Art. no. 104378.

[36] C. Paz, E. Suárez, C. Gil, and J. Vence, "Assessment of the methodology for the CFD simulation of the flight of a quadcopter UAV," *J. Wind Eng. Ind. Aerodynamics*, vol. 218, Nov. 2021, Art. no. 104776.

[37] A. E. Jimenez-Cano, P. J. Sanchez-Cuevas, P. Grau, A. Ollero, and G. Heredia, "Contact-based bridge inspection multirotors: Design, modeling, and control considering the ceiling effect," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3561–3568, Oct. 2019.

[38] S. Ghoury, C. Sungur, and A. Durdu, "Real-time diseases detection of grape and grape leaves using faster R-CNN and SSD mobilenet architectures," in *Proc. Int. Conf. Adv. Technol., Comput. Eng. Sci. (ICATCES)*, 2019, pp. 39–44.

[39] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[40] I. Cheeseman and W. Bennett, "The effect of ground on a helicopter rotor in forward flight," AERADA, London, U.K., Tech. Rep. 3021, 1955.

[41] P. Sanchez-Cuevas, G. Heredia, and A. Ollero, "Characterization of the aerodynamic ground effect and its influence in multirotor control," *Int. J. Aerosp. Eng.*, vol. 2017, pp. 1–17, Aug. 2017.

[42] K. Nordberg, "*Introduction to Representations and Estimation in Geometry*. Linköping, Sweden: Linköping Univ. Electron. Press, 2018, p. 399.

[43] M. Rabah, A. Rohan, S. A. S. Mohamed, and S.-H. Kim, "Autonomous moving target-tracking for a UAV quadcopter based on fuzzy-PI," *IEEE Access*, vol. 7, pp. 38407–38419, 2019.

[44] D. Hartman, K. Landis, M. Mehrer, S. Moreno, and J. Kim. (Jun. 9, 2014). *Quadcopter Dynamic Modeling and Simulation (QUAD-SIM) V1.00 Github Repository*. [Online]. Available: https://github.com/dch33/Quad-Sim

[45] A. Chovancová, T. Fico, L. Chovanec, and P. Hubinsk, "Mathematical modelling and parameter identification of quadrotor (a survey)," *Proc. Eng.*, vol. 96, pp. 172–181, Jan. 2014.