A Dynamic Architecture to Control Multi-Rotors Using Hand Gestures

Nabeel Hussain, Hassan Yousuf, Syed Atif Mehdi*, Kanwal Atif
Faculty of Information Technology & Computer Science, University of Central Punjab, Lahore.
***Correspondence**. syed.atif@ucp.edu.pk

Citation | Hussain. N, Yousuf. H, Mehdi. S. A, Atif. K, "A Dynamic Architecture to Control Multi-Rotors Using Hand Gestures", IJIST, Vol. 6 Issue. 3 pp 1370-1385, Sep 2024
DOI | https://doi.org/10.33411/ijist/20246313701385
Received | Aug 16, 2024 **Revised |** Sep 03, 2024 **Accepted |** Sep 12, 2024 **Published |** Sep 14, 2024.

Traditional methods for controlling multi-rotors typically involve joysticks, radio controllers, and mobile applications. However, these methods pose significant challenges, particularly for novice users like farmers, due to the extensive training and understanding required to effectively operate a copter. This paper introduces a highly adaptable architecture designed to offer an end-to-end solution for controlling a copter using hand gestures. The proposed system leverages a depth sensor and Convolutional Neural Network (CNN) to recognize hand gestures, utilizing a custom dataset collected from both indoor and outdoor environments. Through a series of simulations with novice users, the system has demonstrated successful operation in real-world scenarios. Currently, the architecture can accurately recognize six distinct gestures with an average accuracy of 90.5% across three different test environments with varying lighting conditions. Key features of this proposed solution include its adaptability, reliable performance, especially in low-light conditions, and its user-friendly design, making it particularly well-suited for farmers and other inexperienced users.

Keywords. UAV, Hand Gestures, Human Drone Interaction, Deep Learning, Tensor Flow, CNN, Control Architecture.

Introduction.

In contemporary times, copters have become widely utilized for various purposes globally, including photography/videography, surveillance, and delivery tasks [1][2][3]. One of the rapidly growing fields benefiting from copters is agriculture. These devices offer a safe and effective means of inspecting areas that are otherwise challenging to access, such as wild forests or expansive agricultural fields. The applications of copters in agriculture are diverse, including crop monitoring, planting, and spraying. These applications can potentially enhance crop yield, reduce overall costs related to human resources and time, and improve crop quality. However, many users are inexperienced with copters, facing difficulties with takeoff, landing, obstacle avoidance, and safe navigation. Natural human interaction, which facilitates intuitive and efficient control with minimal learning, could significantly improve usability in such rapidly evolving applications [4]. A gesture-controlled copter could simplify the flying process considerably. Hand gestures are a common form of non-verbal communication, especially useful for hearing-impaired individuals, and are increasingly employed in Human-Computer Interaction (HCI) to enhance automated system usability. Recognizing hand gestures involves various factors and complexities affecting their accuracy and reliability. Vision-based methods for gesture recognition utilize specialized image processing techniques to address specific challenges related to image characteristics.

Conversely, deep learning approaches such as logistic regression, convolutional neural networks (CNNs), XG Boost, Support Vector Classifiers (SVCs), Stochastic Gradient Descent Classifiers (SGDCs), and Transformer adapt to various aspects of hand gestures, including skin tone and system environment, through supervised learning. Particularly, CNN models excel in pattern recognition in images, making them suitable for identifying hands, faces, and other objects. CNNs do not require feature extraction during training and are invariant to scaling and rotation, which enhances their effectiveness for hand gesture recognition [5][6][7]. They are well-suited for handling the spatial complexities involved in recognizing different gestures. However, combining CNNs with other models like Recurrent Neural Network (RNN) or Long Short-Term Memory (LSTM) can enhance performance by incorporating temporal information, especially in dynamic gesture recognition scenarios. Each model has its strengths, but CNNs remain a robust choice due to their ability to directly process and interpret visual data, making them a preferred approach for gesture recognition tasks.

Gestures can be categorized into static and dynamic types. Static gestures involve a single hand pose in one frame, while dynamic gestures encompass sequences of hand poses across multiple frames. Dynamic gesture recognition provides more comprehensive information but is computationally expensive and complex due to its reliance on temporal data [8]. Dynamic gestures are used for continuous control, such as navigation and altitude adjustment. However, they require sophisticated tracking and processing capabilities to ensure smooth and accurate control. It can be complex and requires robust algorithms to differentiate intended gestures from accidental movements. e.g. using the hand upwards and downwards to control the copter altitude or right and left movement to give direction to the copter. On the other hand, static gestures are generally used for discrete commands like takeoff, landing, or hovering. They are simpler to implement and can be very intuitive, improving user experience. However, they require precise recognition and calibration to avoid misinterpretation. Gesture-controlled copters, or drones, represent an exciting advancement in agricultural setup. Many factors contribute to their significance and why they might be preferable to traditional methods.

Technical Complexity.

Traditional controllers or digital interfaces can be complex, with multiple buttons and joysticks that require a steep learning curve. This can be challenging for users who are not technologically confident. Various copter modes, such as loiter control, guided mode, stabilize mode, and remote control, require professional expertise for effective maneuvering. For

instance, operating a copter with a radio controller necessitates knowledge of pitch, yaw, roll, and the functions of various switches, which are not intuitive and require considerable effort to master.

Responsiveness.

Controllers and voice commands can introduce delays. Voice commands can be affected by ambient noise, accents, and misinterpretations, making them less reliable in noisy environments like farms. Additionally, they might not offer the precision needed for complex tasks.

Ease of Use.

Natural interaction methods, being universally accessible and easy to grasp, could simplify copter control, particularly for novice users such as farmers. Enabling farmers to monitor their fields using hand gestures with copters could greatly enhance agricultural efficiency.

Reduced Physical Strain.

Holding and maneuvering a traditional controller for extended periods can be physically demanding. Gesture controls can be less tiring, as they allow for a more natural and less physically strenuous interaction with the drone. The concept of controlling drones through hand gestures has gained attention following the introduction of DJI's Spark (2017) and Mavic 3 Enterprise (2020), which feature gesture control capabilities. Despite their innovation, these drones are expensive and primarily focus on palm detection and limited gesture functions for capturing photos or videos within a short range. In agricultural contexts, however, the ability to operate drones at longer distances and utilize the drone's camera for monitoring, rather than for tracking the farmer, is crucial.

Literature Review.

Research has explored hand gestures for natural human-robot interaction. One approach involves using a LEAP motion sensor [9] for controlling copters through hand gesture detection. However, this sensor has limitations, including a short effective range (30 − 60 cm) and poor performance in bright outdoor environments [10]. Another study proposed a framework for controlling a copter with an onboard camera to detect hand gestures. Results indicated a significant drop in gesture recognition accuracy when the copter was more than 3 feet away from the user, rendering this framework impractical for agricultural settings where the drone's camera is needed for crop monitoring [10]. Machine learning algorithms, such as AdaBoost and Haar wavelet features, have been used to classify hand gestures with a low-resolution webcam [11]. While these methods can recognize 24 static gestures, they are not directly applicable to controlling copters. Data gloves have also been employed to detect hand gestures, achieving 88% accuracy for 26 static gestures [12]. However, data gloves are impractical for farmers due to issues with comfort, durability, and sensitivity varying with hand size.

Hand gestures have also been used to control TV operations. Studies have demonstrated that users can manage a TV using a fixed camera with different hand gestures and hand shapes, but the gestures must be performed within a specific rectangular region to differentiate between hand and face skin color [13]. Another approach involves using depth data from motion and hand location to control a TV [14]. This method involves artificially designed gestures that may be challenging for users to learn and understand.

Voice command control of copters has been explored [15], but it is not universally effective due to voice recognition inaccuracies. Hand gesture control devices offer a more accessible alternative. Research has shown that a car robot can be controlled with hand gestures, achieving 92.2% accuracy in simulated environments, although real-world applications have not been extensively reported [16]. A gesture-controlled robot manipulator system has been developed [17], employing accelerometers and gyroscopes to estimate arm joint angles, but it may not perform well under harsh conditions with vibrations and shocks.

Table 1. Comparative analysis of existing approaches of gesture recognition for copter control

Approach	Strengths	Weaknesses	Application for Copter Control
Vision-Based	• High accuracy with proper lighting and resolution. • Can recognize a wide range of gestures. • Does not require additional hardware.	• Sensitive to lighting conditions. • Computationally intensive. • Requires a clear line of sight.	Effective for complex gestures, but may struggle in low-light or high-motion environments.
Depth-Based	• Better performance in varying lighting conditions. • Can capture 3D gestures. • - Less sensitive to occlusions.	• Requires depth sensors (e.g., Kinect). • Limited range of motion capture. • More expensive hardware.	Useful for capturing gestures with depth information, potentially improving precision.
IMU-Based	• High responsiveness and low latency. • Works in various environmental conditions. • Relatively low cost.	• Limited to gestures that involve motion. • Can be affected by drift and noise. • Less intuitive for complex gestures.	Ideal for quick, intuitive control gestures, but may not handle complex commands well.
Sensor Glove-Based	• High accuracy for hand and finger movements. • Can detect fine gestures. • Good for detailed control.	• Requires wearable equipment. • May be uncomfortable for extended use. • Limited to gestures that involve wearing the glove.	Best for precise and detailed control, but can be cumbersome and less flexible.
EMG-Based	• Can detect muscle movements even without direct line of sight. • Allows for a wide range of control options.	• Requires sensors to be attached to the body. • Can be influenced by muscle fatigue. • May have higher setup complexity.	Useful for gestures involving muscle contractions, but may be less intuitive and require training.
RFID-Based	• Can work in various environmental conditions. • Good for identifying specific hand positions.	• Requires additional tags or sensors. • Limited to predefined gestures. • Tags can be misplaced or obstructed.	Suitable for gesture recognition with specific tagged positions but lacks flexibility for complex gestures.
Hybrid Systems	• Combines strengths of multiple approaches. • Can improve accuracy and robustness. • Adaptable to various scenarios.	• More complex to implement. • Higher cost and resource requirements. • Integration can be challenging.	Offers the best of multiple approaches, but requires careful balancing and integration.

A wearable sensor suite for UAV control has been introduced [18], focusing on mechanomyography and an inertial measurement unit to capture multimodal command signals using convolutional neural networks. However, this system requires individual calibration, which may not be feasible in many scenarios [19] Another study has proposed gesture-based natural language interfaces for defining UAV trajectories but demonstrated only 41% accuracy [20]. Human-machine interaction research has explored dynamic gesture control for vehicles, achieving 95% accuracy for three poses, though some inaccuracies persist due to gesture similarity. This study used a camera mounted on a vehicle roof for gesture recognition [21].

To address touch screen limitations, an infrared camera mounted on the ceiling has been proposed, achieving 96% accuracy in static scenarios, though accuracy drops to 87% in dynamic contexts due to gesture similarity. This system handles both sunlight and nighttime conditions using various segmentation models [22]. The Myo Armband, which detects electromyography for controlling home appliances, offers a unique approach but causes user discomfort and fluctuates in accuracy due to variations in arm fat tissues [23]. An IMU with a complementary filter algorithm on a Raspberry Pi, combined with an MPU6050 sensor and a wearable glove, has been used to navigate a DJI Tello drone, but only roll and pitch movements have been achieved [24].

Deep neural network-based solutions for gesture recognition have been investigated, with models designed using PointNet architecture and depth data from time-of-flight sensors. Comparisons with 2D images and 3D point cloud datasets indicate that 3D depth information significantly enhances gesture recognition accuracy [25]. TensorFlow and deep learning techniques offer efficient real-time gesture recognition, utilizing built-in libraries like Mediapipe, TensorFlow, and Keras. However, the use of a drone-mounted camera presents challenges in open fields where the vehicle may travel beyond the farmer's range [26]. Table 1 summarizes the comparison of various gesture recognition approaches in the context of copter control. The proposed system's architecture has been designed to remain flexible while providing a comprehensive solution to the challenge of controlling a copter through hand gestures (Figure. 1).

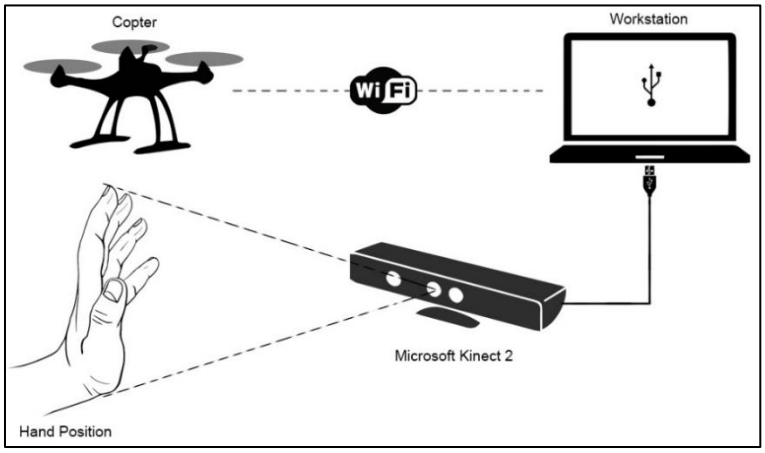

Figure 1. Conceptual diagram of hand gesture-controlled copter

Key research contributions of the paper can be summarized as.

- The proposed system recognizes and classifies hand gestures from live video feeds in real time and translates them into predefined copter operations.
- The system offers a standardized approach to both input and output channels, making it compatible with any RGBD input device for gesture acquisition.
- The core innovation of this architecture lies in its flexibility, allowing for the integration of various modules to provide different functionalities. For example, modules such as

the LEAP motion controller, Kinect, or stereo vision can be easily incorporated to enhance gesture recognition accuracy and motion detection.

Material and Methods.

Investigation Site.

Surveys were conducted with farmers in Muridke Tehsil, near Lahore, Pakistan. The farmers were queried about the practicality of using a gesture-controlled copter for tasks such as field observation, spraying, and fungus detection. The feedback indicated a preference for hand gesture control over traditional controllers and mobile applications, primarily due to a lack of professional knowledge [27]. The proposed system was extensively tested in diverse environments and lighting conditions, both in simulations and real field settings.

This study divides the challenge of controlling copters using hand gestures into four key modules. (i) Gesture Acquisition, (ii) Gesture Recognition, (iii) Communication, and (iv) Control (Figure. 2).

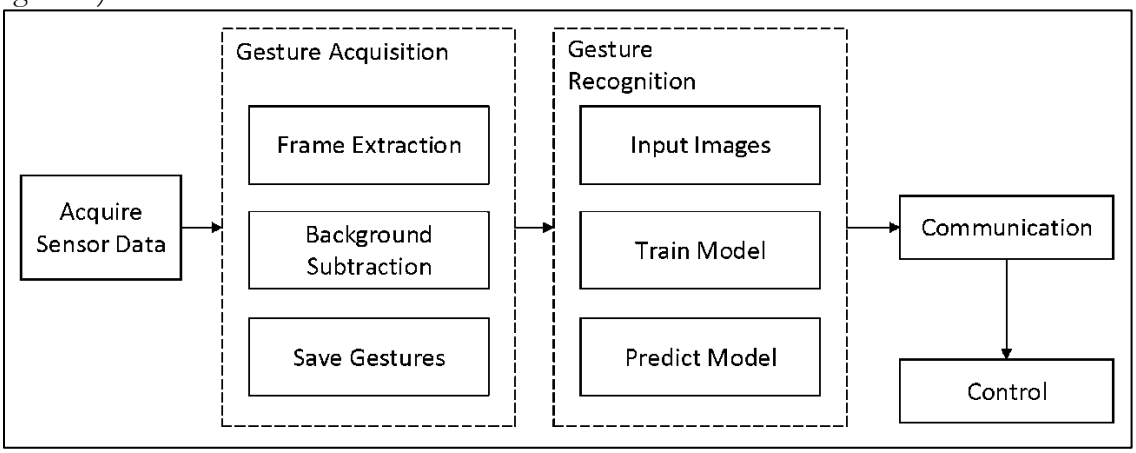

Figure 2. Overview of the proposed architecture

Gesture Acquisition.

The architecture we designed imposes no restrictions on the choice of sensor system for capturing images, as it supports a wide range of RGB and depth cameras. Any depth camera can be employed for gesture capture. Given the advantages of vision-based sensors over alternatives like data gloves or LEAP motion sensors [28], we opted for the Kinect V2 for data acquisition. The Kinect V2 is both cost-effective and widely used in research due to its high-resolution RGB imagery and depth-sensing capabilities [29]. Being one of the most acknowledged sensors, it can efficiently handle issues like complex background and variation in illumination [30]. The Kinect's depth sensor integrates an IR projector with a CMOS sensor to capture video data in various lighting conditions [31].

Dataset Compilation.

Existing datasets for hand gesture recognition often have limitations, such as capturing only 2D images indoors or using special-colored gloves for hand identification [32][33]. To address these shortcomings, we compiled our dataset, which includes both 2D images and 3D depth information, to ensure functionality across different lighting conditions, including daylight and nighttime. We collected a total of 24,000 images of six distinct gestures using Kinect V2. These gestures were recorded in both indoor and outdoor settings to account for the varying lighting conditions encountered by farmers (Figure. 3). To enrich the dataset, different hand sizes and skin colors have been incorporated as well. Images acquired through Kinect have been annotated using the Labeling tool [34]. The background has been eliminated from each frame to prepare them for further processing.

Collecting data in outdoor environments was essential due to the potential overexposure of Kinect's IR camera in sunlight, which can impact image quality. Incorporating depth

information into our dataset helped mitigate this issue. For integrating Kinect with our gesture recognition program, we used libfreenect2, a driver that supports RGB image transfer, IR and depth image transfer, and the registration of RGB and depth images. This setup requires a USB 3.0 controller for powering Kinect.

Gesture Recognition.

Accurate hand gesture detection is a critical component of this architecture, as it must accommodate variations in hand size and skin color among different users. Several algorithms and frameworks exist for hand gesture detection, including Haar features, skin color labeling [35], and image clustering using K-means [36]. The study in [37] addresses gesture recognition in cluttered backgrounds and varying lighting conditions through skin detection and posture contour analysis using Principal Component Analysis.

(a) Outdoor environment **(b)** Indoor environment

Figure 3. Gesture collection in different environments

Unlike traditional machine learning algorithms like AdaBoost, which enhance the performance of simple, weak learning models, our approach leverages Convolutional Neural Networks (CNNs) implemented using TensorFlow, an open-source library developed by Google. TensorFlow operates on a computational graph where nodes represent mathematical operations and edges denote the data flow between these nodes. A notable feature of TensorFlow is its model parallelism capability, allowing different portions of the graph to be trained simultaneously on multiple devices. TensorFlow also supports distributed training, accommodating both similar and diverse devices for various parts of the computational graph [38]. These features enhance the capability to handle large models and datasets, speed up training, and ensure scalable deployment. TensorFlow also provides built-in tools for data augmentation, which is crucial for gesture recognition. Techniques like rotation, scaling, and flipping can help in creating more diverse training data, improving the model's robustness. It also offers visualization tools to monitor the training process and model architecture. This helps in diagnosing issues and improving the model's performance.

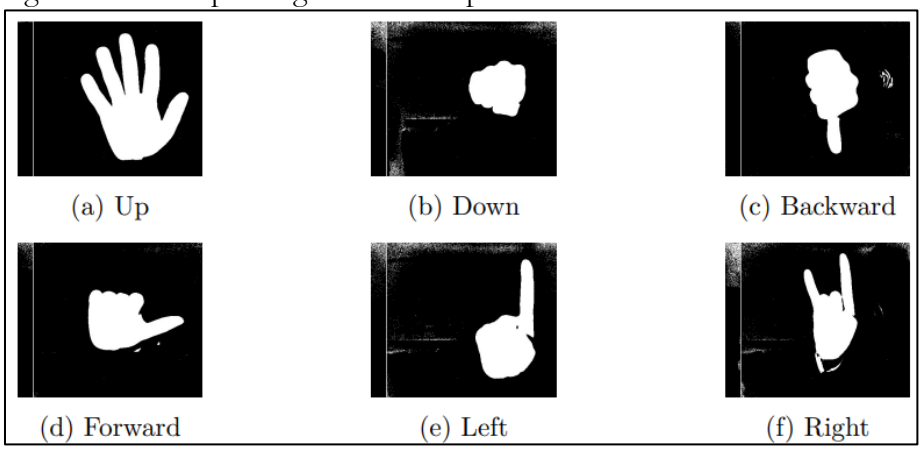

Figure 4. Six gestures of the proposed architecture (a) Open hand towards the camera (b) Punch (c) Closed hand with the thumb pointing downwards (d) Closed hand with the thumb pointing right (e) Closed hand with index finger (f) Closed hand with index and pinky finger.

In our system, depth images are utilized for background removal and feature extraction, followed by gesture recognition, like the methods described in [39]. The system has been trained to identify six distinct gestures, as illustrated in Figure 4. Each gesture corresponds to a specific movement direction for the drone.

Communication.

The communication between the Odroid XU4 and Pixhawk is facilitated using an FTDI (USB to TTL) serial cable. The Odroid XU4, a single-board computer equipped with an Exynos 5422 Octa ARM Cortex-A15 @ 2.0 GHz quad-core processor, is positioned beneath the flight controller to maintain the drone's center of gravity. The architecture employs User Datagram Protocol (UDP) for communication due to its capability to provide low-latency connections essential for real-time copter control [40][41]. UDP also offers a flexible protocol that supports various input devices. UDP is chosen over TCP or Bluetooth in scenarios where low latency is crucial, and the application can handle or tolerate occasional data loss. Its minimal overhead and lack of built-in reliability mechanisms make it suitable for real-time applications. In this setup, the UDP client (the workstation) sends commands to the Odroid, which acts as a server, forwarding these commands to the Pixhawk via its telemetry port.

The system operates in an iterative loop; upon receiving a gesture, the copter executes the corresponding movement and then awaits the next command. If a new gesture is detected while the copter is executing a previous command, the current gesture is overridden, and the copter responds to the new gesture. In the event of a lost connection or Wi-Fi outage during flight, the copter is programmed to return to its launch point immediately. The small packet size in this configuration minimizes the likelihood of timeouts and multiple retransmissions [42]. The system does not impose restrictions on the communication medium, though alternatives like Bluetooth and GSM are less feasible due to their limited range and bandwidth constraints. Wi-Fi, while having a short range, can be extended using multiple wireless access points to cover larger areas.

Control.

Effective control of the copter necessitates precise monitoring of its stability, latency, and movement. Each gesture introduces a latency of 634 ms, calculated using Equation 1, which accounts for the time required to complete pre-arm checks before the copter can take action. When a gesture is recognized, the copter's stability and movement are evaluated, alongside the strength of the communication link. After executing the intended gesture, stability is reassessed before processing subsequent gestures (Figure. 5).

Latency Calculation.

The latency L between the workstation and the copter is calculated as follows.

$$L = \frac{PkS}{LDR} + \frac{LD}{VF} + \frac{QD}{LDR} \quad (1)$$

Latency is determined by several key factors. The first component is the frame serialization time, which is calculated by dividing the packet size (PkS) of the gesture by the link data rate (LDR) measured in bits per second. This represents the time needed to convert the gesture packet into frames for transmission. The second factor, link media delay, is computed by dividing the distance of the link by the velocity factor (VF), which accounts for the speed of data transmission across the communication medium. Finally, queuing delay, which reflects the time a packet spends waiting in a queue before being processed, is calculated by dividing the queue depth (QD) by the link data rate. The total latency is the sum of these delays, encompassing frame serialization time, link media delay, and queuing delay.

System Implementation and Model Choice.

Initially, the gesture recognition system was implemented using scikit-learn, a popular Python library for machine learning. The algorithm employed was the Random Forest Classifier, an ensemble method that builds several decision tree classifiers on various sub-samples of the

dataset and averages their predictions to enhance accuracy and reduce overfitting. However, scikit-learn's Random Forest Classifier was found to be insufficient for this application due to its lack of depth in handling complex image processing tasks and its tendency to disrupt the spatial locality of images, resulting in reduced accuracy and excessive flickering.

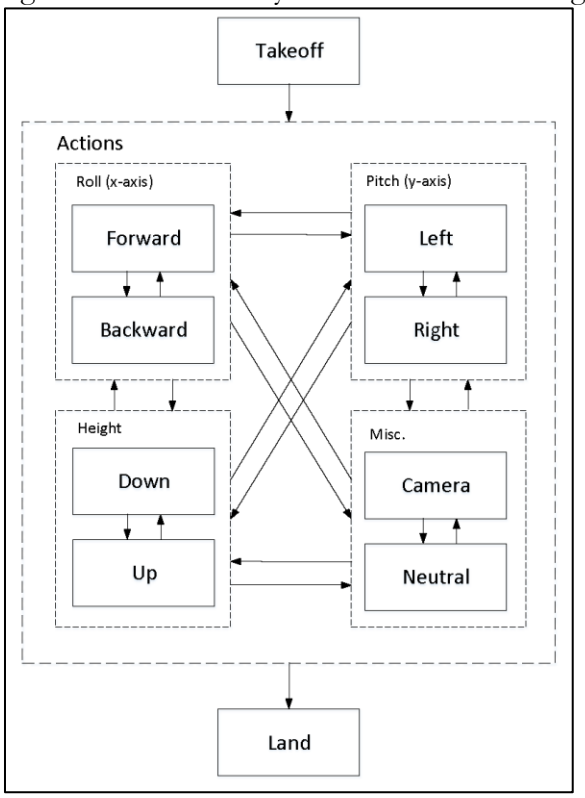

Figure 5. Component state diagram of the system

To overcome these limitations, TensorFlow was subsequently adopted for training the gesture recognition model. This architecture includes seven convolutional layers and seven pooling layers, as depicted in Figure 6. The model was trained using 24,000 images of six different hand gestures, generating a meta file for prediction. In this model, Rectified Linear Units (ReLU) are used as activation functions. ReLU functions convert all negative values to zero while preserving a positive slope for non-negative values, thus avoiding the vanishing gradient problem associated with other activation functions like sigmoid or tanh. This approach enhances the model's ability to accurately recognize gestures by maintaining spatial hierarchies in the image data (Eq. 2).

$$f(x) = max(0, x) \tag{2}$$

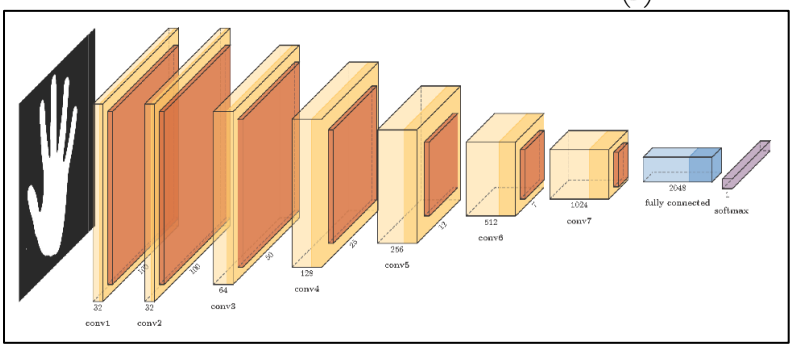

Figure 6. A simple illustration of the Convolutional Neural Network architecture. Tensor Flow uses seven convolutional layers, seven max pool layers, one fully connected layer, and one softmax. The size of the image is indirectly proportional to the depth of the layer.

Rectified Linear Units (ReLU) are employed as the activation function in this architecture due to their simplicity and effectiveness. ReLU transforms all positive values linearly while setting all negative values to zero, offering advantages over other activation functions used in TensorFlow. Following the convolutional layers, a pooling layer is utilized to reduce processing time. Specifically, a max pooling layer is implemented, which reduces the size of the image and thereby curtails computational demands.

To streamline the data further, the tensor is reshaped into a single-dimensional tensor using a flattening layer. This reshaped tensor is then fed into a fully connected layer, which integrates inputs from all previous neurons. The output of this fully connected layer is calculated through matrix multiplication, followed by the addition of a bias offset.

For optimization, the Adam Optimizer is employed to calculate gradients and adjust weights efficiently. The learning rate is set to 0.0001 to minimize the cost function effectively. Finally, the softmax function is applied, as described in Equation 3. This function calculates the probability of each gesture by taking the exponential of each value and then normalizes these values by dividing by the sum of the exponentials. This ensures that the probabilities of all possible gestures sum to 1, providing a clear and normalized output for gesture classification.

$$o(x)_j = \frac{e^{xi}}{\sum_N^{n=1} e^{x_n}} for j = 1...N \quad (3)$$

Experiments and Results.

Safety Mechanism.

Unlike traditional aircraft that rely on cockpit controls, the copter's operation is entirely dependent on telemetry data, which includes GPS coordinates, altitude, heading, and IMU data. To ensure safety and prevent potential crashes, the system calculates the copter's altitude during flight. For instance, if a downward gesture is given when the copter is already 3–4 feet above the ground, the gesture will be ignored to avoid excessive thrust that could damage plants. Similarly, a leftward gesture will cause the copter to move left until another gesture is received. In case a gesture is not recognized, the copter will enter a neutral state, reducing the risk of a crash [43] Additionally, if the GPS fails, the copter can switch to Position Hold (PosHold) mode, allowing for manual control to return it safely. Before arming the copter, several pre-arm checks are performed to ensure a stable flight, including.

Horizontal Dilution of Precision (HDOP). Must be less than 2.0 to ensure a good GPS fix.

- **Magnetic Field Interference.** This should be less than 15% to avoid navigation issues.
- **Battery Charge.** Must be at least 80%. If the battery is low, a failsafe mechanism will trigger, causing the copter to land at its home location.
- **Safety Switch.** Should be off before takeoff to prevent accidental activation of the motors and servos.

Alternatively, these pre-arm checks can be relaxed by increasing the GPS_HDOP_GOOD parameter to 2.2 or 2.5, or by disabling the parameter altogether to take off in a move that does not require GPS (such as Stabilize or AltHold) and switch to Loiter mode after arming. However, this approach is not recommended.

Sensor Selection for Gesture Acquisition.

Various sensor technologies exist which have their strengths and weaknesses for outdoor gesture recognition tasks.

- **LiDAR and Stereo Cameras** offer high precision but may be affected by environmental factors and range limitations.
- **Time-of-flight cameras** provide good performance in varying lighting conditions but can be costly.
- **Radar and Ultrasonic Sensors** are robust in various weather conditions but may lack the resolution needed for detailed gesture recognition.

- **Thermal Cameras** are useful for detecting heat-based gestures but may not provide the detail needed for complex gestures.

Key reasons for selecting Kinect V2 for our project include its low-cost and high-resolution RGB imagery with depth sensing capabilities. Its ability to handle issues like complex background and variation in illumination makes it a good choice for outdoor usage [30].

Gesture Selection.

Designing effective hand gestures for copter control presents a challenge, as the user's experience is closely tied to the gesture's design. Six hand gestures were selected for this study based on the following criteria.

- **Intuitive and Simple.** Gestures should be easy to remember and perform. Complex gestures, while potentially reducing incorrect operations, often lead to poor user experiences.
- **Static.** Static hand gestures are preferred as they reduce system latency and minimize the chance of false recognition. A few well-chosen static gestures are sufficient for the field operations of a copter.
- **No Mutual Intrusion.** Each gesture must be distinct and distinguishable to prevent interference between gestures.

An optimal distance of 1 to 3 feet has been recommended between the user and the Kinect to ensure better gesture recognition.

Simulation Testing and Real-Time Testing.

Simulation Testing.

The proposed system was initially tested in a simulation environment to identify and address potential issues before real-world implementation. This simulation aimed to verify the accuracy of translating gestures into drone movements, thereby protecting the actual copter from potential damage. The simulation was conducted using V-REP, a tool developed by Corpellia Robotics, integrated with the Robot Operating System (ROS). A pre-existing copter model in V-REP (see Figure. 7) was utilized and linked to ROS for this purpose.

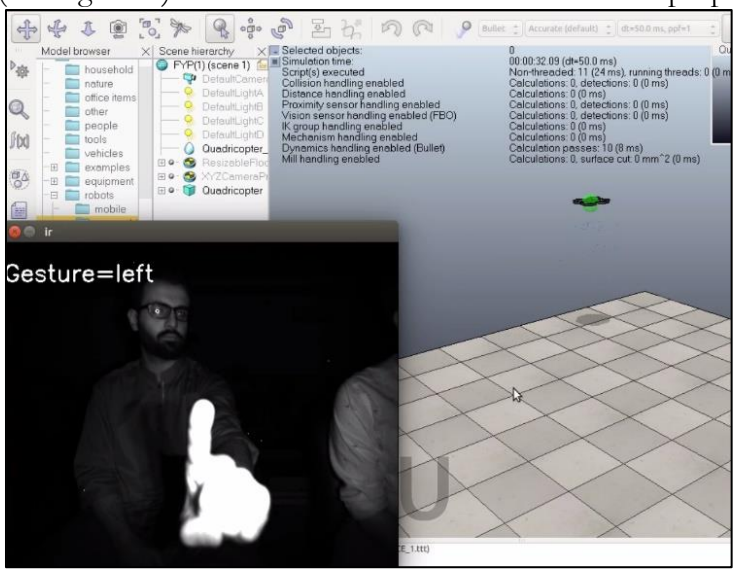

Figure 7. Testing is done in a simulated environment using V-REP and ROS

The simulation testing involved integrating the gesture recognition program with the ROS platform running on Ubuntu. Communication between the simulator and the gesture recognition system was facilitated using ROSBridge, which converts ROS messages into web socket messages. Images captured by the camera were processed and sent to the simulation, where they guided the copter's actions. Following successful simulation trials that covered a range of scenarios, the system was then tested with a real quadcopter.

Real-Time Testing.

For the real-time testing phase, a custom S500 quadcopter was assembled with a Pixhawk flight controller, an Odroid XU4 companion computer, a WiFi module, and an external GPS module. The gesture recognition was performed on a 2.50 GHz CPU running Ubuntu 16.04 LTS. Images collected for training were initially at a resolution of 512x424 but were reduced to 256x212 when using a 2x2 kernel.

The testing involved three environments. indoor, outdoor (day), and outdoor (night), with six test subjects having different hand sizes and colors. For testing, 360 images were specifically excluded from the training dataset to create a confusion matrix (see Table 2). These images were captured across all three testing environments. indoor, outdoor (daylight), and outdoor (nighttime). The accuracy of gesture recognition was slightly reduced in sunlight due to the IR sensor's exposure issues with the Kinect. It has been observed that avoiding the Kinect to be placed directly facing the sun, improves the results.

Table 2. Confusion matrix of all six gestures

		Predicted Values					
		Up	**Down**	**Left**	**Right**	**Forward**	**Back**
Actual Values	**Up**	**55**	0	0	4	0	1
	Down	0	**50**	2	0	6	1
	Left	1	0	**56**	2	0	0
	Right	3	0	0	**51**	0	6
	Forward	0	4	0	0	**54**	2
	Back	0	1	1	0	2	**56**

Detailed results from the experiments in different environments have been summarized in Table 3. Overall, the proposed architecture achieved an accuracy rate of 89.8%. The accuracy varied across environments; indoor testing achieved 95.833%, outdoor (sunlight) testing yielded 90% due to IR interference from sunlight, and outdoor (night) testing achieved 94.166% due to Kinect's effective night vision.

Table 3. Accuracy of six gestures in three different environments

	Indoor			**Sunlight**		**Night**	
	Input images	**Images detected**	**Gesture accuracy (%)**	**Images detected**	**Gesture accuracy (%)**	**Images detected**	**Gesture accuracy (%)**
Up	20	20	100	18	90	18	90
Down	20	19	95	17	85	19	95
Left	20	18	90	18	90	19	95
Right	20	19	95	18	90	18	90
Forward	20	20	100	19	95	19	95
Backward	20	19	95	18	90	20	100
Accuracy			**95.83**		**90**		**94.16**

The processing time for gesture recognition is minimal compared to data collection, ensuring that it does not impact the performance of the recognition algorithm. To enhance accuracy and prevent misclassification, a voting mechanism is employed, where decisions are based on four consecutive frames to confirm a single gesture recognition. It prevents the misclassification of gestures, especially during transition between multiple gestures.

Feedback from farmers indicated a strong preference for controlling the copter through natural hand gestures, highlighting their satisfaction with this mode of interaction compared to traditional controllers or mobile applications. During the field testing with farmers, errors were

noticed when a farmer didn't comply with the required distance which is 12 – 36 cm from the Kinect. It also becomes problematic when farmers tend to deviate from the Kinect's field of view. However, it briefly gets better with practice. Currently, the system has been tested with only six subjects in real-time experiments, extensive field trials can further assist in the improvement of the system's effectiveness.

Conclusion.

This paper presents a robust method for controlling a copter using hand gestures, demonstrating an average accuracy of 90.5% across different environments. The framework incorporates a voting mechanism that uses four consecutive frames to reduce misclassification. The safety mechanisms have prevented crashes, even with inexperienced users. The system's flexibility allows for the replacement of Kinect with other sensors and the adaptation of different gesture recognition algorithms. The gesture recognition module can be updated with future libraries that offer improved hand gesture recognition capabilities. Future work will focus on integrating more gestures, including dynamic ones, for advanced maneuvering and camera control, as well as exploring autonomous path planning.

Acknowledgment. Our sincere thanks to the German Academic Exchange Service (DAAD) for their "German-Pakistani Research Collaborations" program.

Author's Contribution. N. Hussain. Resources, Methodology, Software; H. Yousuf. Data curation, Validation, Writing; S. A. Mehdi. Conceptualization, Supervision, Funding acquisition, Project administration; K. Atif. Writing, Review & editing.

Conflict of Interest. The authors have no conflicts of interest to declare that are relevant to the content of this article.

Project Details. This research has been conducted as part of DeFFAR project (Ref. Number. 57345707) that has been funded by the German Academic Exchange Service (DAAD).

References.

[1] X. Wang, A. Chowdhery, and M. Chiang, "Networked Drone Cameras for Sports Streaming," Proc. - Int. Conf. Distrib. Comput. Syst., pp. 308–318, Jul. 2017, doi. 10.1109/ICDCS.2017.200.

[2] A. Sehrawat, T. A. Choudhury, and G. Raj, "Surveillance drone for disaster management and military security," Proceeding - IEEE Int. Conf. Comput. Commun. Autom. ICCCA 2017, vol. 2017-January, pp. 470–475, Dec. 2017, doi. 10.1109/CCAA.2017.8229846.

[3] V. Gatteschi et al., "New Frontiers of Delivery Services Using Drones. A Prototype System Exploiting a Quadcopter for Autonomous Drug Shipments," Proc. - Int. Comput. Softw. Appl. Conf., vol. 2, pp. 920–927, Sep. 2015, doi. 10.1109/COMPSAC.2015.52.

[4] R. Agrawal and N. Gupta, "Real Time Hand Gesture Recognition for Human Computer Interaction," Proc. - 6th Int. Adv. Comput. Conf. IACC 2016, pp. 470–475, Aug. 2016, doi. 10.1109/IACC.2016.93.

[5] Y. S. Tan, K. M. Lim, and C. P. Lee, "Hand gesture recognition via enhanced densely connected convolutional neural network," Expert Syst. Appl., vol. 175, p. 114797, Aug. 2021, doi. 10.1016/J.ESWA.2021.114797.

[6] S. Bhushan, M. Alshehri, I. Keshta, A. K. Chakraverti, J. Rajpurohit, and A. Abugabah, "An Experimental Analysis of Various Machine Learning Algorithms for Hand Gesture Recognition," Electron. 2022, Vol. 11, Page 968, vol. 11, no. 6, p. 968, Mar. 2022, doi. 10.3390/ELECTRONICS11060968.

[7] T. Begum, I. Haque, and V. Keselj, "Deep Learning Models for Gesture-controlled Drone Operation," 16th Int. Conf. Netw. Serv. Manag. CNSM 2020, 2nd Int. Work. Anal. Serv. Appl. Manag. AnServApp 2020 1st Int. Work. Futur. Evol. Internet Protoc. IPFutu, 2020, [Online]. Available. https://ieeexplore.ieee.org/document/9269056

[8] Y. Liu, M. Dong, S. Bi, D. Gao, Y. Jing, and L. Li, "Gesture recognition based on Kinect," 6th Annu. IEEE Int. Conf. Cyber Technol. Autom. Control Intell. Syst. IEEE-CYBER 2016, pp. 343–347, Sep. 2016, doi. 10.1109/CYBER.2016.7574847.

[9] A. Sarkar, K. A. Patel, R. K. G. Ram, and G. K. Capoor, "Gesture control of drone using a motion controller," 2016 Int. Conf. Ind. Informatics Comput. Syst. CIICS 2016, Apr. 2016, doi. 10.1109/ICCSII.2016.7462401.

[10] K. Natarajan, T. H. D. Nguyen, and M. Mete, "Hand gesture controlled drones. An open source library," Proc. - 2018 1st Int. Conf. Data Intell. Secur. ICDIS 2018, pp. 168–175, May 2018, doi. 10.1109/ICDIS.2018.00035.

[11] T. B. Dinh, V. B. Dang, D. A. Duong, T. T. Nguyen, and D. D. Le, "Hand gesture classification using boosted cascade of classifiers," Proc. 4th IEEE Int. Conf. Res. Innov. Vis. Futur. RIVF'06, pp. 139–144, 2006, doi. 10.1109/RIVF.2006.1696430.

[12] S. A. Mehdi and Y. N. Khan, "Sign language recognition using sensor gloves," ICONIP 2002 - Proc. 9th Int. Conf. Neural Inf. Process. Comput. Intell. E-Age, vol. 5, pp. 2204–2206, 2002, doi. 10.1109/ICONIP.2002.1201884.

[13] A. Shimada, T. Yamashita, and R. I. Taniguchi, "Hand gesture based TV control system - Towards both user - & Machine-friendly gesture applications," FCV 2013 - Proc. 19th Korea-Japan Jt. Work. Front. Comput. Vis., pp. 121–126, 2013, doi. 10.1109/FCV.2013.6485473.

[14] X. Liu and K. Fujimura, "Hand gesture recognition using depth data," Proc. - Sixth IEEE Int. Conf. Autom. Face Gesture Recognit., pp. 529–534, 2004, doi. 10.1109/AFGR.2004.1301587.

[15] S. Supimros and S. Wongthanavasu, "Speech recognition - Based control system for Drone," Proc. 2014 3rd ICT Int. Sr. Proj. Conf. ICT-ISPC 2014, pp. 107–110, Oct. 2014, doi. 10.1109/ICT-ISPC.2014.6923229.

[16] X. H. Wu, M. C. Su, and P. C. Wang, "A hand-gesture-based control interface for a car-robot," IEEE/RSJ 2010 Int. Conf. Intell. Robot. Syst. IROS 2010 - Conf. Proc., pp. 4644–4648, 2010, doi. 10.1109/IROS.2010.5650294.

[17] K. Root and R. Urniezius, "Research and development of a gesture-controlled robot manipulator system," IEEE Int. Conf. Multisens. Fusion Integr. Intell. Syst., vol. 0, pp. 353–358, Jul. 2016, doi. 10.1109/MFI.2016.7849513.

[18] Y. Ma et al., "Hand gesture recognition with convolutional neural networks for the multimodal UAV control," 2017 Work. Res. Educ. Dev. Unmanned Aer. Syst. RED-UAS 2017, pp. 198–203, Nov. 2017, doi. 10.1109/RED-UAS.2017.8101666.

[19] O. M. Sincan and H. Y. Keles, "AUTSL. A large scale multi-modal Turkish sign language dataset and baseline methods," IEEE Access, vol. 8, pp. 181340–181355, 2020, doi. 10.1109/ACCESS.2020.3028072.

[20] M. Chandarana, E. L. Meszaros, A. Tmjillo, and B. Danette Allen, "Analysis of a gesture-based interface for UAV flight path generation," 2017 Int. Conf. Unmanned Aircr. Syst. ICUAS 2017, pp. 36–45, Jul. 2017, doi. 10.1109/ICUAS.2017.7991390.

[21] M. Zobl, M. Geiger, B. Schuller, M. Lang, and G. Rigoll, "A real-time system for hand gesture controlled operation of in-car devices," pp. III–541, May 2004, doi. 10.1109/ICME.2003.1221368.

[22] F. Parada-Loira, E. Gonzalez-Agulla, and J. L. Alba-Castro, "Hand gestures to control infotainment equipment in cars," IEEE Intell. Veh. Symp. Proc., pp. 1–6, 2014, doi. 10.1109/IVS.2014.6856614.

[23] P. J. Gonzalo and A. Holgado-Terriza Juan, "Control of home devices based on hand gestures," 5th IEEE Int. Conf. Consum. Electron. - Berlin, ICCE-Berlin 2015, pp. 510–514, Jan. 2016, doi. 10.1109/ICCE-BERLIN.2015.7391325.

[24] A. Budiyanto, M. I. Ramadhan, I. Burhanudin, H. H. Triharminto, and B. Santoso,

"Navigation control of Drone using Hand Gesture based on Complementary Filter Algorithm," J. Phys. Conf. Ser., vol. 1912, no. 1, p. 012034, May 2021, doi. 10.1088/1742-6596/1912/1/012034.

[25] R. Mirsu, G. Simion, C. D. Caleanu, and I. M. Pop-Calimanu, "A PointNet-Based Solution for 3D Hand Gesture Recognition," Sensors 2020, Vol. 20, Page 3226, vol. 20, no. 11, p. 3226, Jun. 2020, doi. 10.3390/S20113226.

[26] B. Latif, N. Buckley, and E. L. Secco, "Hand Gesture and Human-Drone Interaction," Lect. Notes Networks Syst., vol. 544 LNNS, pp. 299–308, 2023, doi. 10.1007/978-3-031-16075-2_20.

[27] I. Kagirov, D. Ivanko, D. Ryumin, A. Axyonov, and A. Karpov, "TheRuSLan. Database of Russian Sign Language." pp. 6079–6085, 2020. Accessed. Sep. 12, 2024. [Online]. Available. https://aclanthology.org/2020.lrec-1.746

[28] J. P. Sahoo, A. J. Prakash, P. Pławiak, and S. Samantray, "Real-Time Hand Gesture Recognition Using Fine-Tuned Convolutional Neural Network," Sensors 2022, Vol. 22, Page 706, vol. 22, no. 3, p. 706, Jan. 2022, doi. 10.3390/S22030706.

[29] M. Oudah, A. Al-Naji, and J. Chahl, "Elderly Care Based on Hand Gestures Using Kinect Sensor," Comput. 2021, Vol. 10, Page 5, vol. 10, no. 1, p. 5, Dec. 2020, doi. 10.3390/COMPUTERS10010005.

[30] D. Sarma and M. K. Bhuyan, "Methods, Databases and Recent Advancement of Vision-Based Hand Gesture Recognition for HCI Systems. A Review," SN Comput. Sci., vol. 2, no. 6, pp. 1–40, Nov. 2021, doi. 10.1007/S42979-021-00827-X/FIGURES/18.

[31] R. A. El-Laithy, J. Huang, and M. Yeh, "Study on the use of Microsoft Kinect for robotics applications," Rec. - IEEE PLANS, Position Locat. Navig. Symp., pp. 1280–1288, 2012, doi. 10.1109/PLANS.2012.6236985.

[32] G. Moon, S. I. Yu, H. Wen, T. Shiratori, and K. M. Lee, "InterHand2.6M. A Dataset and Baseline for 3D Interacting Hand Pose Estimation from a Single RGB Image," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 12365 LNCS, pp. 548–564, 2020, doi. 10.1007/978-3-030-58565-5_33.

[33] "LSA64. A Dataset for Argentinian Sign Language · Facundo Quiroga." Accessed. Sep. 12, 2024. [Online]. Available. https://facundoq.github.io/datasets/lsa64/

[34] C. P. Papageorgiou, M. Oren, and T. Poggio, "General framework for object detection," Proc. IEEE Int. Conf. Comput. Vis., pp. 555–562, 1998, doi. 10.1109/ICCV.1998.710772.

[35] C. Quan and J. Liang, "A Simple and Effective Method for Hand Gesture Recognition," pp. 302–305, Jun. 2017, doi. 10.1109/ICNISC.2016.072.

[36] M. Panwar, "Hand gesture recognition based on shape parameters," 2012 Int. Conf. Comput. Commun. Appl. ICCCA 2012, 2012, doi. 10.1109/ICCCA.2012.6179213.

[37] N. H. Dardas and E. M. Petriu, "Hand gesture detection and recognition using principal component analysis," IEEE Int. Conf. Comput. Intell. Meas. Syst. Appl. Proc., pp. 11–16, 2011, doi. 10.1109/CIMSA.2011.6059935.

[38] A. Parvat, J. Chavan, S. Kadam, S. Dev, and V. Pathak, "A survey of deep-learning frameworks," Proc. Int. Conf. Inven. Syst. Control. ICISC 2017, Oct. 2017, doi. 10.1109/ICISC.2017.8068684.

[39] K. K. Biswas and S. K. Basu, "Gesture recognition using Microsoft Kinect," ICARA 2011 - Proc. 5th Int. Conf. Autom. Robot. Appl., pp. 100–103, 2011, doi. 10.1109/ICARA.2011.6144864.

[40] I. Coonjah, P. C. Catherine, and K. M. S. Soyjaudah, "Experimental performance comparison between TCP vs UDP tunnel using OpenVPN," 2015 Int. Conf. Comput. Commun. Secur. ICCCS 2015, Jan. 2016, doi. 10.1109/CCCS.2015.7374133.

[41] T. Le et al., "Reliable user datagram protocol for airborne network," Proc. - IEEE Mil. Commun. Conf. MILCOM, 2009, doi. 10.1109/MILCOM.2009.5380007.

[42] M. Petrovic and M. Aboelaze, "Performance of TCP/UDP under ad hoc IEEE802.11," 10th Int. Conf. Telecommun. ICT 2003, vol. 1, pp. 700–708, 2003, doi. 10.1109/ICTEL.2003.1191496.

[43] S. R. Haque, R. Kormokar, and A. U. Zaman, "Drone ground control station with enhanced safety features," 2017 2nd Int. Conf. Converg. Technol. I2CT 2017, vol. 2017-January, pp. 1207–1210, Dec. 2017, doi. 10.1109/I2CT.2017.8226318.