Pakistan Journal of Engineering Technology and Science (PJETS)
Volume 13. Issue.2, PP. 01-11, December 2025
https://doi.org/10.22555/pjets.v13i2.1265
E-ISSN 2224-2333
P-ISSN 2222-9930

A Comparative Case Study to Control Drones using Hand Gestures

Hassan Yousuf [1], Nabeel Hussain [1], Kanwal Atif [1], Syed Atif Mehdi [1*]
Faculty of Information Technology and Computer Science,
University of Central Punjab, Lahore, Pakistan

[*]Corresponding author: syed.atif@ucp.edu.pk

Abstract:

As human-computer interfaces are constantly evolving day by day, hand gesture recognition (HGR) systems are becoming reliable and cost-effective. The interfaces for natural interaction allow users to convert their gestures into commands for a computer system. Multirotor drones are usually controlled using radio controllers, which require good experience and training. It is difficult for novice users to properly handle the multirotor using such controllers. This paper builds upon our previously developed architecture for drone control via hand gestures [1]. While the initial work demonstrated the feasibility of using a vision sensor, this study expands its scope by exploring sensor-based and vision-based approaches of HGR. It has been done by integrating and testing three hand gesture data acquisition devices. The aim is to evaluate the performance and usability of these sensors under varying conditions, including indoor and outdoor environments. Experiments, initially conducted in simulation and then with a real drone, show that such devices can be used to train novice users to maneuver the multirotor. Results reveal significant differences in accuracy and practicality, providing actionable insights for selecting sensors based on application needs. This comparative study highlights the adaptability of the existing architecture and its potential for future advancements in human-computer interaction.

Keywords: *Drone, Hand gesture recognition, Microsoft Kinect, Myo Armband, Intel RealSense*

I. INTRODUCTION

Development of drones conventionally focuses on technical application aspects, for instance, computer vision and artificial intelligence, ranging from a copter's hardware functionality to system software. The field of human-robot interaction already rigorously analyzes the utility of computer vision technology for drones and its functionality. However, this type of research does not focus on the experience of interacting with gesture-based control. Natural interaction is not only easy to communicate but also a universal way that is comprehensible for everyone, especially for farmers and novice users. Drones are widely used in agriculture for many purposes, including spraying and crop monitoring, but one of the challenges that persists to this day is controlling the copter. Novice users, such as farmers, are inexperienced with copters, and getting their controlling skills up to the mark takes a lot of effort and work. They have difficulty controlling a copter as it is a complex task using a radio controller. Default controlling mechanisms require a well-built skill and cannot be learned simply and intuitively.

With the escalating reliability of robotic systems, many human responsibilities are now being carried out by robots. The rising level of such systems increases efficiency and safety while simultaneously decreasing the human workload. An initial survey of different farmers has been carried out [1]. They were asked about flying a quadcopter using different control mechanisms and inspecting their crops with a camera mounted on it. Results suggest that most farmers prefer natural interaction as it is more suitable and easier to use, compared to complex hand controllers. Moreover, an external sensor will be required to detect hand gestures while the camera on the quadcopter will perform the inspection.

This work uniquely contributes to the field by providing an in-depth comparison of three different gesture-sensing devices to control the drones, emphasizing their usability for novice users in practical applications. The methodology

Creative Common CC BY: This article is distributed under the terms of the Creative Commons Attributes 4.0 License.
It permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

involves minimal modifications to the existing control system [1], ensuring its compatibility with three different sensors, hence exploring its potential for future developments in human-computer interaction and autonomous control. Each sensor was tested under identical conditions to evaluate gesture recognition accuracy, latency, and usability. Comparative metrics include accuracy in gesture detection, environmental adaptability, and ease of deployment. The experimental setup for each device is described in subsequent subsections.

The rest of the paper is structured in the following sections. Section II discusses the related work done with gesture-sensing devices. Section III gives an overview of the proposed architecture. Section IV describes each gesture-sensing device in detail and the TensorFlow model. Section V discusses the results of different devices and their accuracies, and Section VI concludes the paper by giving an outlook on future work.

II.LITERATURE REVIEW

A significant amount of research has been done in recognizing hand gestures as a trivial part of Human-Computer Interaction, being a natural and nonverbal medium. Research has been done in both vision-based [2, 3] and sensor-based approaches [4] to recognize the hand gestures. These applications are then used to control robots in diverse scenarios. In [5], the authors discuss their work on communicating with TV using hand gestures. A camera is embedded at the top of the TV, which detects gestures in a specific region. It is not only a user-friendly approach, but it also gets rid of the controller. Users just need to perform gestures to control the TV. Home automation is one of the areas where a lot of work has been done these days. The authors in [6][7] describe controlling the home devices using hand gestures. This kind of approach uses an armband to control home appliances. This work is unique; however, it irritates users to wear the armband for a longer period of time. Its accuracy can fluctuate because users have different fat tissues in their arms, which generate different electrical values. Natural interaction is also necessary for safety purposes. In [8][9], the authors have used hand gestures to control multimedia in a car. The camera is mounted on the car's roof and detects the driver's gestures while driving. Aside from multimedia, a lot of work has been done with depth-sensing devices to recognize human gestures. The authors in [10] use Microsoft Kinect to recognize gestures. In this work, eight dynamic gestures are trained on a Support Vector Machine (SVM) using depth images with an accuracy of 84.6%. The same approach has been used in [11], where the authors recognize gestures and use both skeleton tracking and depth images. Although the accuracy has increased slightly in this approach, human overlapping is one of the challenges in this approach, which deteriorates the accuracy as well. The authors in [12] describe the control of a copter using a LEAP motion sensor. However, the problem with this approach is that it has a short range of 60 cm, and it does not work well in sunlight. Authors have evaluated multiple input devices for hand gesture recognition to use in games and found Leap Motion and Kinect to be equally usable and efficient in the gaming experience [13].

A framework has been proposed to control the drone using an on-board camera to detect hand gestures in [14]. According to the performed experiments, the accuracy of gesture recognition considerably drops if the drone moves more than 3 feet away from the user, which is not feasible in agricultural fields. [15] has implemented machine learning algorithms such as AdaBoost and Haar wavelet features to classify hand gestures using a low-resolution webcam. Although they can recognize 24 static gestures, they are not translated to control the copter and require complex image processing. Similarly, [16] has recognized 26 static hand gestures using data gloves with an accuracy of 88%. However, wearing a data glove is not practical as it is difficult to carry and will produce a lot of sweat, which is uncomfortable for farmers in agricultural fields. Moreover, the sensitivity of sensors depends on the hand size, which varies a lot among farmers. Hand posture and gesture recognition using the Myo Armband and spectral collaborative representation-based classification have been discussed in [17]. The accuracy achieved with this algorithm is 97%; however, they have only discussed static gestures, which is not feasible if the idea is to increase the number of gestures. Similarly, [18] has proposed a gesture recognition system for the game of Hand Cricket. The author [19] has proposed data collection using the Myo Armband for American Sign Language recognition. The data collection was done with eight channels from the band and used the Myo server function from MATLAB, which effectively rendered the data collection.

In [20], hand gesture recognition is done with Intel RealSense, in which two types of input are given to the machine learning model. One is the RGB image, and the other is the depth image. The accuracy is 99.4% which uses a total of 168,000 images, consisting of 84,000 RGB images and 84,000 depth images for training. With the increase in research on using hand gestures to control the drones, efforts have been made to develop specialized flight controllers that support the control of drones through hand gestures. One such study is providing an efficient control system for drones using a hybrid model consisting of deep learning and reinforcement learning, incorporating hand gestures [21][22]. Moreover, [23] has demonstrated a low-cost gaze and fixation tracker by capturing eye surface geometry using Intel

RealSense. Possible applications of such a technique can be the detection of faces or hands being observed in autism studies or the detection of the quadrant being observed on a computer screen.

While various gesture-sensing technologies have been explored, challenges such as environmental adaptability and user-friendliness persist. This reinforces the necessity for the current study, which aims to provide a deep analysis and comparison for future researchers in selecting a robust and versatile solution capable of addressing these issues.

III. PROPOSED METHODOLOGY

A. Problem Description

Controlling drones using conventional radio controllers presents significant challenges for users who are not experienced operators, such as farmers and novice users. These controllers require a high level of coordination and skill set, often leading to frustration and inefficient operation. The complexity involved in maneuvering drones with traditional methods poses barriers to adoption and effective use, especially in agricultural or fieldwork settings where users seek simple and intuitive solutions. The work aims to evaluate these aspects by experimenting with an adaptable gesture-based control system that allows for straightforward, user-friendly interaction, reducing the learning curve and enabling effective drone operation without extensive prior training [1]. Three sensor systems will be deployed to confirm the adaptability of the system as well as compare their performance with respect to hand gesture identification for controlling the drone in real time.

B. Scope of Study

This study aims to investigate the performance of three hardware solutions, namely Microsoft Kinect V2, Intel RealSense D435, and Myo Armband, that allow the user to provide hand gestures on the ground station while the drone may fly long distances and carry out the intended agricultural tasks in the field. The specialized control architecture, as per Figure 1, will receive this information and translate it into accurate drone movements. The features of the control architecture and the optimization of gesture detection were covered in the previous work [1] and are outside the scope of the current analysis.

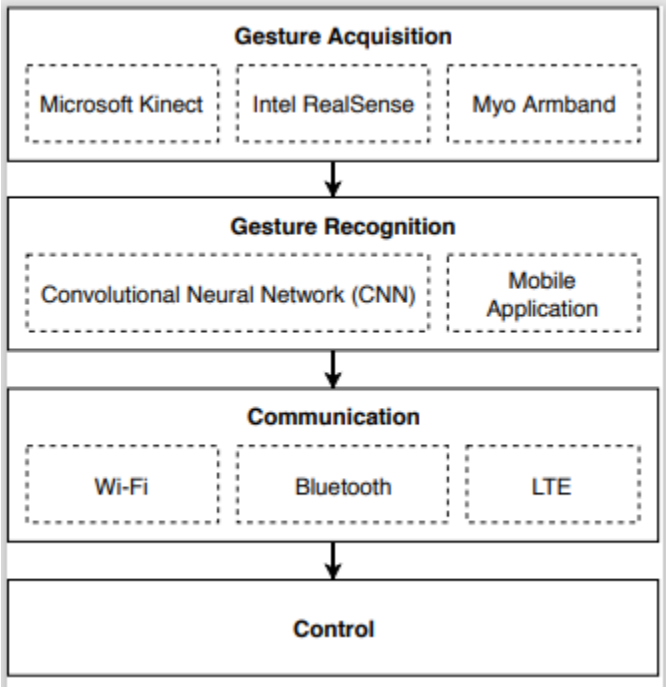

Figure 1: Flow diagram of the evaluation system

C. Solution Framework

The architecture illustrated in Figure 1 is designed to streamline the process from gesture detection to drone response. It is divided into four main phases: Gesture Acquisition, Gesture Recognition, Communication, and Control. Each phase plays a critical role in ensuring a seamless flow of data from the user's input to the drone's execution of commands. The Gesture Acquisition phase captures hand movements using selected sensors, such as the Microsoft Kinect V2, Intel RealSense D435, or Myo Armband. The Gesture Recognition phase employs machine learning algorithms implemented in TensorFlow to identify specific gestures and translate them into commands. In the Communication phase, these commands are transmitted using UDP over Wi-Fi to ensure low-latency data transfer. The Control phase involves the drone's response, supported by a robust communication link and pre-arm safety checks. This structured approach ensures that gestures are accurately interpreted and effectively translated into the desired drone actions, allowing for a more intuitive and adaptable control system.

A. Gesture Acquisition

The devices discussed in this paper used for acquisition are Microsoft Kinect V2, Myo Armband, and Intel RealSense D435. In Figure 2, the collection of gestures with Microsoft Kinect and Intel RealSense is depicted, highlighting data captured under different lighting conditions. This comparison underlines the adaptability of these devices for reliable gesture recognition in both controlled indoor and variable outdoor environments.

Figure 2: Gesture collection with Microsoft Kinect [1] and Intel RealSense in different environments with varying lighting conditions

Technical specifications for Microsoft Kinect V2, Intel RealSense D435, and the Myo Armband are detailed in Table 1, covering aspects like hardware compatibility, dimensions, and power requirements. This comparative view helps assess the operational strengths and situational constraints each device presents. Kinect and RealSense are depth sensor devices, whereas Myo is an armband that provides electromyography signals (EMG) and Inertial Measurement Unit (IMU) values. Kinect has a depth, IR, and RGB camera, due to which it works at nighttime as well. The main problem with Kinect is that it does not work well in sunlight because sunlight has IR waves that interfere with Kinect's IR camera. Moreover, Kinect needs an external power supply, which affects its portability in fields. Intel RealSense D435 is a depth camera with a maximum depth range of up to 10 meters, depending on the lighting conditions. The main advantage of this camera is that it has a built-in Vision Processor D4. Additionally, it does not need any external power supply. It just requires USB 3.0, which makes it superior to Kinect.

Table 1: Comparison between Microsoft Kinect V2, Intel RealSense D435 and Myo Armband specifications

	Microsoft Kinect V2	**Intel RealSense D435**	**Myo Armband**
Hardware Compatibility	USB 3.0	USB 3.0	USB 2.0
Dimensions	12" x 3" x 2.5"	7" x 2" x 1.5"	5.2" x 3.2" x 4.5"
Weight	970 g	72 g	255 g
Power Supply	DC	USB	3.7V Li-Po, 220 mAh
Power Consumption	12 watts	<3.5 watts	N/A
Range (m)	0.5 – 4.5	0.2 – 10	Bluetooth
Field of View	57º x 43º	87º x 58º	N/A

Myo Armband is a gesture-sensing device that can detect hand gestures by reading the electrical activities of the human muscle. The armband comprises eight stainless steel sensors capable of getting raw EMG data from the muscle.

The frequency of EMG data is 200 Hz, and the range of this data is between −128 to 128. It has a built-in IMU sensor, which has a frequency of 50 Hz. It is a rechargeable device with a long battery life of around 48 hours, making it easy to carry to almost any area.

B. Gesture Recognition

The method implemented for gesture recognition in this architecture uses a Convolutional Neural Network (CNN) using TensorFlow, which is an open-source library by Google. TensorFlow is based on a computational graph where nodes represent mathematical operations and edges represent the flow of data between nodes. An important feature of TensorFlow is model parallelism, where different portions of the graph can be trained on multiple devices in parallel. TensorFlow can also train on a distributed environment as it can employ similar and variable devices for training various parts of the graph and the depth image is used for background removal, feature extraction and then used to recognize gestures. The gesture detection technique and its optimization have been described here [1] in detail.

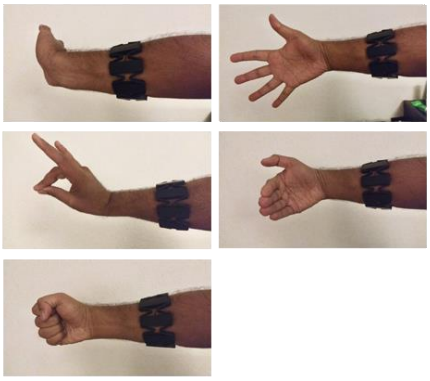

Figure 3: Five pre-defined gestures of Myo Armband

Kinect and RealSense are trained on machine learning algorithms whereas with Myo Armband there are five pre-defined gestures (Figure 3). However, additional gestures can be added through EMG and IMU fusion. An android application has been developed for this architecture which acquires EMG and IMU values and stores them in a CSV file for further analysis.

C. Communication

Currently, the communication medium implemented in this architecture is the User Datagram Protocol (UDP) over Wi-Fi. Although UDP is a connectionless protocol and not reliable, it is good to send short messages over a network. It is faster than other protocols because error recovery is not applied. Odroid is connected to the flight controller using an FTDI serial cable. Odroid XU4 is a single-board computer that uses an Exynos 5422 Octa ARM Cortex-A15 @ 2.0 GHz quad-core. Architecture constantly looks for gestures. Once a gesture is identified, the copter will perform according to it and wait for the next one. If a new gesture is performed while the copter is completing the last gesture, then the current gesture is dropped, and a new gesture will be executed. If there is no Wi-Fi or the copter loses connection during flight, it will return to launch immediately. Since the packet size in this scenario is small, it can reduce the probability of timeout and multiple re-transmissions. Two communication mediums are used for the Myo Armband because Myo acquires gestures from an Android application and supports Bluetooth only. Once the data is transferred to the Android application, it identifies the gesture and sends a command over Wi-Fi to the copter's companion computer for further processing. There is no restriction on the choice of communication medium between Bluetooth and GSM. However, Bluetooth has a very short and limited range, while GSM has a bandwidth lag, which would not be feasible in this scenario. Moreover, Wi-Fi has a short range but can be easily extended by using multiple wireless access points around the area. Additionally, the use of 5G can also be implemented in this architecture. The salient features of 5G network include work with reliable connectivity with zero percent latency, high-speed data transfer, able to sustain a broad range of user equipment, scalable to various devices, and support data demands in voice communication, web access, and multimedia data.

D. Control

To control the copter, it is essential to calculate its stability, latency, and movement. After every gesture given to the system, there is a latency of 634ms before and after the copter can fly because certain pre-arm checks need to be fulfilled every time a new gesture is given. If a user gives the right gesture, the copter's stability and movement must be checked, along with how strong the communication is, before performing the right gesture. After the correct gesture is completed, the stability of movement is rechecked before another gesture is worked out. A custom S550 quadcopter with a flight controller, external GPS module, and a companion computer with Wi-Fi is built from scratch to achieve full control over the copter and to make it programmable as well. Pixhawk is the flight controller in this scenario, and Odroid XU4 is our companion computer. Unfortunately, Odroid does not come with a built-in Wi-Fi module. Hence, a TP-LINK 300mbps Wi-Fi dongle has been used, which supports IEEE 802.11n/g/b.

IV. EXPERIMENTAL SETUP

This section gives details about Kinect, Myo, and RealSense. The TensorFlow model, different safety mechanisms, and system setup are discussed. Before TensorFlow, the gesture recognition system was implemented using scikit-learn, a Python library for image processing. Random Forest Classifier was implemented at the initial stages, which was a meta estimator that fits many decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The reason to use TensorFlow later instead of scikit-learn was that the Random Forest Classifier does not have depth and destroys the locality of the image. Convolutional Neural Networks are implemented in TensorFlow, which gives full control over the model.

A. Microsoft Kinect

In Microsoft Kinect, 24,000 gestures were collected to get maximum accuracy during prediction. These six gestures were collected in both indoor and outdoor environments because most of the farmers have to operate in different lighting conditions. The gestures were collected in different balances, i.e., 4000 sequences per gesture class. Multiple gestures were available online in different sizes, but no gestures were collected in harsh weather conditions. These datasets have been collected in rooms, in daylight, and at nighttime. Collecting data in outdoor environments is crucial because the IR camera of Kinect overexposes the frame, which can also affect the result of the depth camera. In agricultural fields, most farmers have to face sunlight; hence, it is vital to get relevant training datasets. In this scenario, Kinect's IR and depth sensor is used to capture gestures. Moreover, the RGB frame rate of Kinect is not fixed and may drop to 15 fps (from 30 fps) in low light conditions. After acquiring the dataset from Kinect, some pre-processing is performed on this data, which will eliminate the background from each frame, as shown in Figure 4.

Figure 4: Background subtraction on Microsoft Kinect's depth camera's image

B. Intel RealSense

Intel RealSense D435 is another device for gesture acquisition that provides 90 fps and has a built-in processor. There is no need for external power, and it uses only USB 3.0. The model trained on Kinect can also be worked with RealSense because it provides depth disparity with much better results. Dataset is also collected with RealSense, indoor and outdoor, and is balanced. Additionally, RealSense can gather more precise results than Kinect V2 at distances less than 1m. It has been observed that the precision results for the D435 are more scattered than Kinect. Moreover, RealSense offers various parameters for adapting to the scene. Our experiments used default presets for high-accuracy measurements without adaptation to the current lighting and materials. The main difference between the Intel RealSense D435 and the Microsoft Kinect V2, aside from the extreme difference in form factor, is that the D435 is equipped with a stereo camera, whereas the Kinect does not have it. Therefore, in environments with extreme lighting

conditions, Kinect V2 performs poorly due to interference from sunlight. RealSense can still manage to compute effective depth values.

C. Myo Armband

In this architecture, EMG and IMU data are used to classify gestures. The Myo Armband should be worn at the widest part of the arm to get an accurate and precise rating of the pulses. IMU provides values to check the position and angle of the arm with the help of accelerometer, gyroscope, and magnetometer for orientation. With the help of these values, static gestures are now converted to dynamic gestures. For training, around 50 users were asked to perform different

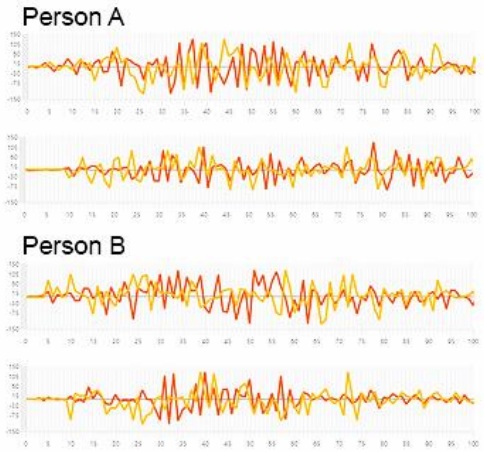

Figure 5: Raw EMG of two persons posing gestures

gestures, as each person has different fat tissue and size. The effect of these factors on the accuracy of identification of correct hand gestures is evident from Table 2. The main difference between Myo and other gesture-sensing devices is that Myo detects gestures in a continuous stream using EMG signals. The gesture has a start and end point, which can be detected on a continually monitored signal, but these gestures might change, and this problem can even occur for the same user. Along with inter-individual and intra-individual differences, the time length of different gestures in a set might also show variations. In [24], the authors have already researched increasing the number of predefined gestures from five to nineteen. The researchers concluded that more gestures could confuse the user since the difference between gestures is small. Additionally, experienced users performed better than non-experienced users. For classification, simple scikit-learn algorithms were used for Myo Armband i.e., Random Forest Classifier.

D. TensorFlow Model

This experimental setup uses TensorFlow for training, which consists of seven convolution layers and seven pooling layers. In this model, 24,000 images are trained, and a .meta file is generated for the prediction. In the convolutional layer, rectified linear units (ReLU) are linear in which all the negative values are converted to zero. The slope, however, does not create a plateau or saturate when x gets large. It does not have a vanishing gradient problem, suffered by other activation functions like sigmoid or tanh. The definition of the ReLU max function is described in Equation 1.

$$f(x) = max(0, x) \qquad (1)$$

Despite other activation functions in TensorFlow, ReLU is linear(identity) for all positive values, and zero for all negative values. After the convolutional layer, the pooling layer is used to reduce the processing time. In this architecture, the max pool layer is implemented, which reduces the image size and curtails the processing time.

Table 2: Comparison of Myo Armband with different fat tissues

Gesture	Orientation	Less Fat	Normal Fat	More Fat	Accuracy	Pose
Up	Up	6	5	5	88.8%	
Down	Down	6	5	5	88.8%	
Left	N/A	5	4	4	72.2%	
Right	N/A	5	5	4	77.7%	
Forward	Up	5	4	4	72.2%	
Backward	Down	5	4	4	72.2%	
Total		88.8%	75%	72.2%	78.7%	

By using the flattening layer, the tensor is simply reshaped to create a single-dimensional tensor. A fully connected layer will receive input from all previous neurons in the previous layer. The output of this layer is computed by matrix multiplication followed by bias offset. Adam Optimizer is used for gradient calculation and weight optimization. The cost is minimized with a learning rate of 0.0001. Finally, softmax Equation 2 is applied, which provides the probability of each gesture and normalizes the values by taking the exponential of each value x, divided by the sum of the exponential values. This step ensures that the sum of the components of the output vector is 1.

$$o(x)_j = \frac{e^{xi}}{\sum_N^{n=1} e^{x_n}} for j = 1...N \qquad (2)$$

V. DISCUSSION & RESULTS

The performance of the three sensors varied significantly under different environmental conditions. Microsoft Kinect exhibited a high accuracy of 87% in indoor settings but struggled with sunlight interference due to its reliance on IR technology. Intel RealSense outperformed Kinect in outdoor environments at 90.2%, offering superior depth sensing and better adaptability to lighting variations. Myo Armband, while portable and versatile, showed reduced accuracy due to inter-individual variability in EMG signals. These results highlight the importance of sensor selection based on application-specific needs, such as portability, accuracy, and environmental constraints. Confusion matrices for both gesture-sensing devices are shown in Table 3 (a) and (b).

There are several reasons for this difference; one main reason is that RealSense provides better results in sunlight as compared to the Kinect and has precision at a distance of less than 1m. A total of 540 images that were not part of the training set were tested in both cases, and the results show correctly recognized gestures with Kinect and with RealSense. Since Kinect provides 30 fps and RealSense provides 90 fps, a voting mechanism is implemented in this architecture, which waits for five consecutive frames to prevent any misclassification, which can be used to average the predictions of sub-labels for any new predictions.

Although the Kinect can give better results in indoor environments, it is not easy to carry such heavy hardware in outdoor environments and look for a power source. The user is bound to a specific location for flight operations. Kinect V2 may work under sunlight, but the result will deteriorate tremendously due to overexposure. Intel's RealSense provides 90 fps, which is much better than Kinect's 30 fps. It also has a processor, which provides on-board processing and does not put much load on the machine.

Table 3: Confusion matrix of all six gestures with

(a) Microsoft Kinect

Actual Values \ Predicted Values	Up	Down	Left	Right	Fwd	Back
Up	75	2	2	6	4	1
Down	1	79	0	3	4	3
Left	4	0	78	2	2	4
Right	2	1	5	77	3	2
Fwd	0	2	3	1	81	3
Back	0	2	3	0	5	80

(b) Intel RealSense

Actual Values \ Predicted Values	Up	Down	Left	Right	Fwd	Back
Up	82	0	3	3	0	2
Down	0	81	1	2	5	1
Left	2	2	80	4	1	1
Right	4	0	5	76	2	2
Fwd	0	4	1	0	83	2
Back	0	1	1	0	3	85

The difference in both cases is that these experiments were conducted in sunny environments; however, Kinect V2 does not work well under sunlight because sunlight overexposes Kinect's IR frame. Due to this, the accuracy of Kinect drops tremendously. On the other hand, RealSense has no problem detecting gestures in sunny environments as it offers various parameters for adapting to the scene. While Kinect V2 is good in indoors and at nighttime, RealSense performs significantly better indoors, outdoors, as well as in sunlight, which is why RealSense's accuracy is much better than Kinect V2, as shown in Table 4. When implementing the voting mechanism, RealSense gives more flexibility when checking for consecutive frame results.

Table 4: Comparison between Microsoft Kinect and Intel RealSense after applying the voting mechanism

		Microsoft Kinect V2		Intel RealSense D435		
Gesture	**Input Images**	**Images Detected**	**Accuracy**	**Images Detected**	**Accuracy**	**Pose**
Up	25	24	96%	25	100%	
Down	25	23	92%	25	100%	
Left	25	24	96%	24	96%	
Right	25	25	100%	24	96%	
Forward	25	22	88%	23	92%	
Backward	25	23	92%	25	100%	
Total			94%		97.3%	

Myo Armband provides much ease for users to carry as compared to other devices. With the armband, users just need to wear it on the broadest part of the arm and connect with an Android application. However, Myo has some limitations; for instance, its accuracy is not as good as other gesture sensing devices due to the differences in fat tissue in human arms. Figure 5 shows two raw EMG signals from two users. They are making the same gestures, i.e., fingers spread and fist. This is because each person has different fat tissue and signals generated from the first gesture may be nearly identical to the signals produced from someone else's wave in/out. Furthermore, wearing the armband for a

long time is also an issue because it produces sweat and will get uncomfortable over time. Dynamic gestures are easy to implement in Myo because of its IMU, which can be fused with EMG easily, unlike other gesture-sensing devices.

VI. CONCLUSION & FUTURE WORK

This study provides a comparative analysis of three distinct gesture-sensing devices for drone control, highlighting their strengths and limitations. While Microsoft Kinect is ideal for controlled indoor environments, Intel RealSense proves more effective in outdoor settings due to its superior adaptability to lighting conditions. Myo Armband offers portability but requires further refinement to address its accuracy-related issues. It can be used over Bluetooth, which provides some range but becomes uncomfortable to wear after some time and may produce a lot of sweat in warm environments. This study provides two main contributions: First, an analysis of three different gesture-sensing devices to control the multirotor and their comparative results has been evaluated. Second, a robust drone-control architecture [1] has been evaluated for its adaptability and scalability across different sensors. Future work will explore more complex gestures, dynamic maneuvers, and multi-sensor integration to enhance the system's capabilities.

REFERENCES

[1] N. Hussain, H. Yousuf, S. A. Mehdi, and K. Atif, "A Dynamic Architecture to Control Multi-Rotors Using Hand Gestures," International Journal of Innovations in Science & Technology, vol. 6, no. 3, pp. 1370–1385, Sep. 2024.

[2] J. Shin, A. S. M. Miah, M. H. Kabir, M. A. Rahim, and A. Al Shiam, "A Methodological and Structural Review of Hand Gesture Recognition Across Diverse Data Modalities," IEEE Access, vol. 12, pp. 142606–142639, Sep. 2024.

[3] J. Qi, L. Ma, Z. Cui, et al., "Computer Vision-Based Hand Gesture Recognition for Human-Robot Interaction: A Review," Complex Intell. Syst., vol. 10, pp. 1581–1606, 2024.

[4] R. Tchantchane, H. Zhou, S. Zhang, and G. Alici, "A Review of Hand Gesture Recognition Systems Based on Noninvasive Wearable Sensors," Advanced Intelligent Systems, vol. 5, no. 10, 2300207, Jul. 2023.

[5] A. Shimada, T. Yamashita, and R. Taniguchi, "Hand Gesture-Based TV Control System—Towards Both User- and Machine-Friendly Gesture Applications," Proc. 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision, pp. 121–126, Jan. 2013.

[6] P. Gonzalo and A. Holgado-Terriza Juan, "Control of Home Devices Based on Hand Gestures," Proc. 2015 IEEE 5th Int. Conf. Consumer Electronics—Berlin (ICCE-Berlin), pp. 510–514, Sep. 2015.

[7] F. Alemuda and F. J. Lin, "Gesture-Based Control in a Smart Home Environment," Proc. 2017 IEEE Int. Conf. Internet of Things (iThings), GreenCom, CPSCom, and SmartData, pp. 784–791, Jun. 2017.

[8] M. Zobl, M. Geiger, B. Schuller, M. Lang, and G. Rigoll, "A Real-Time System for Hand Gesture-Controlled Operation of In-Car Devices," Proc. 2003 Int. Conf. Multimedia and Expo (ICME), vol. 3, pp. III–541, Jul. 2003.

[9] F. Parada-Loira, E. Gonzalez-Agulla, and J. L. Alba-Castro, "Hand Gestures to Control Infotainment Equipment in Cars," Proc. 2014 IEEE Intelligent Vehicles Symp., pp. 1–6, Jun. 2014.

[10] K. K. Biswas and S. K. Basu, "Gesture Recognition Using Microsoft Kinect®," Proc. 5th Int. Conference Automation, Robotics and Applications, pp. 100–103, Dec. 2011.

[11] Y. Liu, M. Dong, S. Bi, D. Gao, Y. Jing, and L. Li, "Gesture Recognition Based on Kinect," Proc. 2016 IEEE Int. Conf. Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), pp. 343–347, Jun. 2016.

[12] A. Sarkar, K. A. Patel, R. K. G. Ram, and G. K. Capoor, "Gesture Control of Drone Using a Motion Controller ," Proc. 2016 Int. Conf. Industrial Informatics and Computer Systems (CIICS), pp. 1–5, Mar. 2016.

[13] A. S. Khalaf, S. A. Alharthi, A. Alshehri, I. Dolgov, and P. O. T. Dugas, "A Comparative Study of Hand Gesture Recognition Devices for Games," Human-Computer Interaction: Multimodal and Natural Interaction (HCII 2020), LNCS, vol. 12182. Springer, 2020.

[14] K. Natarajan, T. D. Nguyen, and M. Mete, "Hand Gesture Controlled Drones: An Open Source Library," Proc. 2018 1st Int. Conf. Data Intelligence and Security (ICDIS), pp. 168–175, Apr. 2018.

[15] 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. 2006 Int. Conf. Research, Innovation and Vision for the Future, pp. 139–144, Feb. 2006.

[16] S. A. Mehdi and Y. N. Khan, "Sign Language Recognition Using Sensor Gloves," Proc. 9th Int. Conf. Neural Information Processing (ICONIP), vol. 5, pp. 2204–2206, Nov. 2002.

[17] A. Boyali, N. Hashimoto, and O. Matsumoto, "Hand Posture and Gesture Recognition Using Myo Armband," Proc. IEEE 4th Global Conf. Consumer Electronics (GCCE), pp. 200–201, Oct. 2015.

[18] K. S. Krishnan, A. Saha, S. Ramachandran, and S. Kumar, "Recognition of Human Arm Gestures Using Myo Armband for the Game of Hand Cricket," Proc. IEEE Int. Symp. Robotics and Intelligent Sensors (IRIS), pp. 389–394, Oct. 2017.

[19] C. Savur and F. Sahin, "American Sign Language Recognition Using Surface EMG Signal," Proc. IEEE Int. Conf. Systems, Man, and Cybernetics (SMC), pp. 2872–2877, Oct. 2016.

[20] B. Liao, J. Li, Z. Ju, and G. Ouyang, "Hand Gesture Recognition With Generalized Hough Transform and DC-CNN Using RealSense," Proc. 2018 8th Int. Conf. Information Science and Technology (ICIST), pp. 84–90, Jun. 2018.

[21] F. S. Khan, M. N. Haji Mohd, S. A. B. M. Zulkifli, G. E. M. Abro, S. Kazi, and D. M. Soomro, "Deep Reinforcement Learning-Based UAV Control Using 3D Hand Gestures," Comput. Mater. Contin., vol. 72, no. 3, pp. 5741–5759, 2022.

[22] I. E. Al-Shayeb, G. E. M. Abro, F. S. Khan, R. Boudville, and A. M. Abdallah, "Integrating AI-Driven Robust Control Algorithm With 3D Hand Gesture Recognition for QUAV," Proc. IEEE 14th Int. Conf. Control System, Computing and Engineering (ICCSCE), pp. 70–75, 2024.

[23] M. Draelos, Q. Qiu, A. Bronstein, and G. Sapiro, "Intel RealSense = Real Low-Cost Gaze," Proc. 2015 IEEE Int. Conf. Image Processing (ICIP), pp. 2520–2524, Sep. 2015.

[24] Z. Lu, X. Chen, Q. Li, X. Zhang, and P. Zhou, "A Hand Gesture Recognition Framework for Mobile Devices," IEEE Trans. Human-Machine Syst., vol. 44, no. 2, pp. 293–299, Apr. 2014.