IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 4, 2020, accepted November 20, 2020, date of publication December 18, 2020,
date of current version December 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3045858

Smart Glove and Hand Gesture-Based Control Interface for Multi-Rotor Aerial Vehicles in a Multi-Subject Environment

KIANOUSH HARATIANNEJADI AND RASTKO R. SELMIC, (Senior Member, IEEE)
Department of Electrical and Computer Engineering, Concordia University, Montreal, QC H3G 1M8, Canada
Corresponding author: Kianoush Haratiannejadi (k_harati@encs.concordia.ca)

ABSTRACT This paper introduces an adaptable, human-computer interaction method to control multi-rotor aerial vehicles in unsupervised, multi-subject environments. A region-based convolutional neural network (R-CNN) first detects subjects in a frame and their faces' regions of interest (RoIs), which are then fed to a facial recognition module to search for the main subject within the frame. The R-CNN model supplies the right-hand RoI of the main subject to a convolutional neural network (CNN) that classifies the right-hand gesture. A motion processing unit (MPU) and four flex sensors are embedded in the left hand's smart glove to produce discrete and continuous signals. Those signals are generated based on the bending of left-hand fingers and the left hand's roll angle and then fed to a support vector machine (SVM) to classify the left-hand gesture. Three validation layers have been implemented, including a human-based validation, classification validation, and the system validation. The comprehensive experimental results have validated the proposed method.

INDEX TERMS Gesture recognition, pose estimation, unmanned aerial vehicles, validation module, wearable sensors.

I. INTRODUCTION

Wearable sensors and gesture recognition techniques have been used to develop a reliable and acceptable human-robot interaction, including wearable motion sensors for the hearing and speech impaired [1], a wearable gesture recognition glove to control robotic devices in harsh environments [2], and wearable gesture-based gadgets for interaction with mobile devices [3]. An adaptive control-based interface with self-validation for multi-rotor aerial vehicles that recognizes and classifies right and left-hand gestures in a single-subject environment was developed in [4]. The quality of different aspects of life has been increased by using multi-rotor aerial vehicles in recent years, especially in monitoring and transportation [5].

Various tools have been proposed for gesture recognition, from extending mathematical models based on hidden Markov chains [6] to approaches based on 3D cameras [7], and recognizing human activities [8]. In a single-subject environment, it is assumed that the user faces a camera and only gestures with one hand [4]. However, in real-life applications, a camera often captures both hands and more than one subject in a frame.

This paper proposes an adaptable, gesture-based control interface for a multi-rotor aerial vehicle in a multi-subject environment. The main goal here is to recognize and classify the right-hand and left-hand gestures of the subject of interest in a multi-subject environment. This paper considers that the main subject controls the targeted device – a multi-rotor aerial vehicle.

We employed a skeleton detection module to detect subjects in a multi-subject environment. After identifying the subject of interest in the frame, the proposed algorithm detects the main subject's hands. Finally, a CNN classifies the detected right hand's gesture into one of the pre-defined classes.

In order to classify the left-hand gesture, a set of four flex sensors and a motion processing unit (MPU) are implemented on the smart glove to produce discrete and continuous signals based on the bending value of each of the left-hand fingers and the roll angle of the left hand, [4]. The produced signals are fed into an SVM to classify the left hand's gesture.

We verified the proposed algorithm on several groups of two, three, and four subjects to study the algorithm's behavior

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

in various scenarios. We implemented the proposed algorithm on Nvidia Jetson AGX Xavier GPU.

A. CONTRIBUTION

This paper offers the following novelties compared to existing results in [9]–[12]: (i) the smart glove that produces 16 different gestures (commands) including two control signals that are used in the real-time control; (ii) a hand recognition module before gesture classification that improves the classifier performance; (iii) an online learning method that allows the system to be adaptable to new users; and (iv) a classifier self-validation and an overall system validation with a human-in-the-loop for improved system reliability.

Note that the smart glove, which can generate 16 different commands, appends the image processing approach with four gesture-based commands to provide enriched control capabilities compared to the gesture classification alone. For instance, varying the angle in ''turn right'' and ''turn left'' commands is carried out by bending the left-hand fingers, or changing the drone's altitude is carried out by altering the angle of the glove.

This paper is an extension of our conference paper [4] and results presented there. Here we expanded our research results and included: (i) detection of the main subject in a multi-subject environment; (ii) detection of the main subject's right-hand region and classifying the main subject's right-hand gesture; and (iii) a comprehensive set of experiments in multi-subject environments.

II. BACKGROUND AND RELATED WORK

Human-robot interactions have vast applications in control and navigation of quad-rotors [7], control of a vehicle to minimize the user's distraction while driving [13], and more. In [7], [13] one of the limitations is that the user has to be alone in front of the camera while we consider a multi-subject environment where we first recognize the main subject, followed by a hand gesture classification.

In [14], a gesture recognition-based method using depth information from pixels for the human skeleton was proposed, where the depth was used to segment the body into joints. Whereas in this paper, we apply the skeleton detection on 2D images to detect all subjects present in each frame. The skeleton detection module also detects subjects' heads and both hands.

Authors in [15] propose a continuous hand gesture recognition and detection method using radar technology to gather the intermediate frequency signal from the environment and estimate the different features of the hand gesture. While the method in [15] recognizes the hand gestures using the k-means algorithm, our proposed method first identifies the main subject in the environment and then classifies the hand gesture using a CNN model.

A skeleton detection module is used to detect all subjects present in each frame [16] and label their bodies to determine their heads Region of Interest (RoI). A facial recognition module is then applied to identify the main subject among all subjects [17]. The skeleton detection module determines the RoI of the pair of hands of the detected main subject. Finally, a gesture classification module is used to classify the gesture of the right hand of the subject of interest. The system recognizes the left-hand gesture using the smart glove and an SVM module that uses data from flex sensors and the MPU on the glove. It simultaneously classifies right-hand and left-hand gestures in each frame with three validation modules running to improve the hand gesture classification modules' performance.

A. CONVOLUTIONAL NEURAL NETWORK MODEL

The CNN model used in this paper consists of different hidden layers. We also used the dropout technique in two separate layers with sigmoid activation function to avoid overfitting [18]. Based on our application and the dataset, we have chosen the number of CNN hidden layers by increasing the number of layers until the classifier achieved the desired performance. We have chosen three layers of max-pooling, two layers of dropout, five rectified activation functions in different layers, one sigmoid activation function, one flattening layer, and two fully connected layers for the CNN hidden layers.

1) POOLING LAYERS

The pooling layers concept is to decrease the feature maps' resolution and summarize the input distortions in a smaller feature map. We use the max-pooling method in the CNN design to extracts the highlights from the input frame. In max-pooling, we select the largest element within each feature map such that [19]:

$$y_{i,j} = \max_{(p,q)\in R_{i,j}} x_{p,q} \,, \tag{1}$$

where $x_{p,q}$ is defined as

$$x_{p,q} = \sum_{i=1}^{m} \omega_i x_{p,q_i} \,, \tag{2}$$

$y_{i,j}$ is the output of the pooling function associated with the feature map region $R_{i,j}$, and $x_{p,q}$ is the input element at location (p, q) which embodies an area around the position (i, j) in the input frame. In (2), ω_i is the i-th weight associated with x_{p,q_i} which is i-th element of $x_{p,q}$, and m is the number of elements in input $x_{p,q}$.

By using a max-pooling technique, unnecessary information is removed from the generated feature maps in the convolutional layer, thus reducing the map's dimensions. The maximum value in each sub-region of the feature maps is stored, and the remaining values are deleted as unnecessary information. The pooled feature maps are generated to form the max-pooling layer.

2) DROPOUT TECHNIQUE

The dropout technique temporarily disables a neuron at each iteration with a probability of p, which prevents overfitting

during the training process. The dropped-out neurons are tested with probability p at every iteration of training, so a dropped-out neuron at one step can be activated at the next iteration [20].

The dropout is applied to input neurons with a parameter p during the training process, controlling the participation of each neuron x_j with an independent Bernoulli random variable α_j for each step [21]:

$$y_i = \frac{1}{p}\sum_{j=1}^{N} \omega_{j,i}(\alpha_j \cdot x_j), \quad \alpha_j \sim B(1,p), \tag{3}$$

where $B(1,p)$ is the Bernoulli function which generates the independent random variable α_j with probability p. The scaling factor $\frac{1}{p}$ scales the output y_i to keep the expected value of the output during the training process. The weight from input x_j to input x_i is denoted by $\omega_{j,i}$, and N is the number of neurons connected to the neuron x_i.

3) ACTIVATION FUNCTION

Activation function limits the amplitude of the neuron's output and maps the neuron's output to a range such as between 0 and 1 [20], [22].

Authors in [22] compared various activation functions for different classification problems. The comparison reveals the suitable activation functions for training an image processing-based algorithm with the proposed architecture in this paper are a rectified linear activation function (ReLU) and a sigmoid activation function. Moreover, the training process requires fewer iterations with ReLU and a sigmoid activation function than other activation functions to solve nonlinear problems. Additionally, we found that our CNN's accuracy is 5% higher with these two functions than other activation functions such as radial basis function and hyperbolic tangent function.

ReLU is defined by

$$f(x) = max(x, 0), \tag{4}$$

where $f(x)$ is the output of the function and x is the input neuron. We use the sigmoid function that is defined as:

$$g(x_i) = \frac{1}{1+e^{-x_i}}, \tag{5}$$

and input neuron x_i is defined as:

$$x_i = \sum_{j=1}^{N} \omega_j x_{ij}, \tag{6}$$

where ω_j denotes the weight associated to the j-th element of input x_i and N is the number of element in input x_i. In this paper, we apply the sigmoid activation function to obtain each hand-gesture class's probability.

4) OPTIMIZER

The primary purpose of using optimizer is to reduce the losses, increase the learning rate, and provide accurate results [23]. Based on comparisons between adaptive moment estimation (Adam), gradient descent, adagrad, and adadelta in [23], [24] it has been shown that the suitable optimizer for the proposed neural network structure is Adam due to its algorithm speed, computation capacity, and obtained results in the proposed method. The weights update law for Adam optimizer is given in [25]

$$\omega_{t+1} = \omega_t - \frac{\eta \cdot \hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}}, \tag{7}$$

where ω_t and ω_{t+1} are the current model weight and the next iteration model weight, respectively. The learning rate (step size) is η and to avoid division by zero, a small positive term ε is added. The bias correction of the moving averages $\widehat{m}_t$ and $\widehat{v}_t$ in the Adam optimizer are as follows:

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t}, \end{aligned} \tag{8}$$

where m_t and v_t are moving averages

$$\begin{aligned} m_t &= (1-\beta_1)g_t^2 + \beta_1 m_{t-1}, \\ v_t &= (1-\beta_2)g_t^2 + \beta_2 v_{t-1}, \end{aligned} \tag{9}$$

g_t is gradient on current mini-batch, and β_1, β_2 are the exponential decay rates.

B. SMART GLOVE

The proposed smart glove converts hand gestures to a set of pre-defined commands [26], which maps the fingers' dynamic movements to a set of pre-defined surgery robot actions. We use the flex sensors' output values to determine the main subject command to control a multi-rotor aerial vehicle.

In [27], the authors compared the consistency and the level of interaction in wearable and non-wearable sensors to control a robot. They tested different gadgets such as keypads, joysticks, single button, and gloves on random subjects. Their results show that the users were much more comfortable creating commands and controlling other devices using wearable sensors. Hence, the users made more intuitive and natural connections with the hand movements of the wearable sensors.

In [28]–[30], the authors described a vision-based recognition and how to control a machine. They demonstrated that the vision-based gesture recognition techniques are of utmost importance in designing highly efficient and intelligent human-machine interfaces. The users have a natural, creative, and intuitive way of communicating with the target machine. These features make the vision-based interaction with computers as natural as an interaction with another human.

We used a glove with an MPU module, four flex sensors, and an Arduino UNO microprocessor board for the smart glove implementation. The MPU module calculates the left hand's angles in three axes, and the flex sensors measure the bending of each finger on the left hand. The microprocessor gathers the raw data from the MPU and the flex sensors for

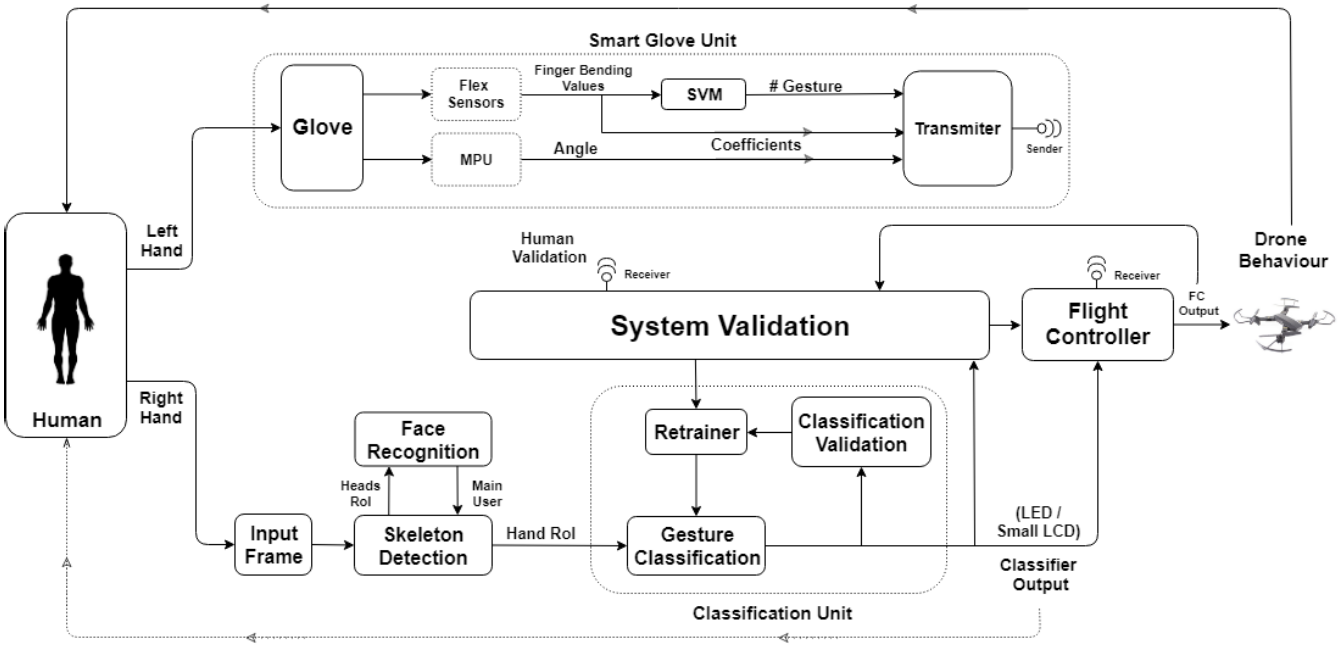

FIGURE 1. System block diagram for multi-subject environment including three separate validation modules: (i) validation of the gesture classifier; (ii) System Validation; and (iii) Human-in-the-loop.

further processing and classifying the left-hand gesture. For instance, when the user makes a pointing gesture, the microprocessor receives the MPU module's angles and each flex sensor's bending value. An SVM model then classifies the collected data into one of the 16 possible left-hand gestures and calculates the coefficients. After all, the microprocessor transmits the classified gesture and the coefficients to the controller unit and the system validation module.

III. DESIGN DETAILS

We developed a gesture-based, closed-loop control system with validation modules for single- and multi-subject environments. The user can validate and disapproves of the system's output through the smart glove by observing the drone behavior [4]. In case the user disapproves the drone's movement, the system validation module monitors the classifier's outputs and the flight controller's outputs (command executioner) to trace the source of a problem. As illustrated in Figure 1, the gesture-based control interface for a multi-subject environment contains three main units and four main modules: (1) a human-in-the-loop for validation, (2) a skeleton detection module, (3) a facial recognition module, (4) a right-hand detection module, (5) a gesture classification unit, (6) the controller unit, and (7) the system validation module.

Two gesture recognition methods are used to interact with an aerial vehicle: (i) image processing-based method and (ii) smart glove-based method.

A. GESTURE RECOGNITION MODULE USING SMART GLOVE

The user calibrates the MPU by keeping his/her hand parallel to the ground and fully closing and opening the fingers before the system starts. After the calibration process, the microprocessor transmits the generated signals from the flex sensors and the MPU to two of the main units, the flight controller and the system validation module, Figure 1.

The smart glove operates between two different modes simultaneously. The first mode considers each finger as "on switch" when the finger is bent and "off switch" when the finger is fully open. The second mode uses the exact bending value of each flex sensor. More details on the smart glove modes are given in [4]. Figure 2 shows various signatures with four flex sensors for each gesture.

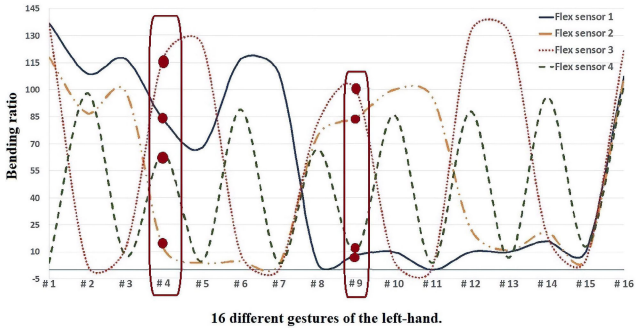

FIGURE 2. Finger bending fluctuation of four flex sensors. These signals are used for left-hand gesture recognition and continuous control signals generation. As indicated in the horizontal axis, a combination of four flex sensors bending values represents a specific gesture signature.

B. SKELETON DETECTION MODULE

In every input frame, the skeleton detection module recognizes the joints (or skeleton) of the subjects [16]. The proposed method is a top-down approach to detect the joints. The initial step in finding subjects in each frame is to find

a nose. The skeleton detection module starts from the nose, and then it searches to find eyes, ears, neck, and then the rest of the body as long as it is visible in the frame. If the skeleton detection module finds the nose and ears, the module counts that as a subject. Afterward, the module connects all detected joints and forms the body of each subject.

In case the skeleton detector fails to detect all the joints of each subject (when subjects are close to the camera, for instance), the skeleton detection module saves the related coordinates of each joint and creates a list of detected joints and their coordinates in the frame with a specific number for each joint.

In the first step, the skeleton detection module receives an input RGB frame and feeds it through a neural network to detect each subject's skeleton. This module's algorithm searches for a similar pattern and calculates the confidence scores, keypoint positions, and keypoint confidence scores from the network for each pattern that it detects. With a gesture confidence score, we can determine the overall confidence in estimating the skeleton (between 0 and 1). This keypoint confidence score can be used to hide the skeletons that do not have a high score and sets the keypoint with a high score in the array that forms the skeleton. After the module finds the detectable keypoints, it connects them and forms the skeleton.

C. FACIAL RECOGNITION MODULE

After the skeleton detection module detects subjects in each frame, the next step is to determine the main subject. Detected head regions by the skeleton detection module are fed to the facial recognition module to determine the main subject. Suppose the facial recognition module confirms the detected face of the main subject. In that case, the algorithm uses the joints coordinates obtained from the skeleton detection module to find the main subject's right hand.

For the facial recognition model, we use the Histogram of Oriented Gradients (HOG) method [31], which searches face patterns in the detected head region and face landmark in order to recognize the main-subject face. The goal of the HOG method is to find a pattern in the image that has a face. HOG finds the basic structure of the image based on the lighting of each pixel. As the next step, finding the part of the image similar to a known HOG pattern extracted from a data set of other training faces is crucial in identifying the main subject's face.

In the first step, the face recognition module uses the HOG method to get the locations and outlines of each person's eyes, nose, mouth, and chin. These locations are needed for the feature extraction for the main subject identification. Then, the module compares the HOG results from the input frame with the provided picture from the main subject, and if results are above (or below) a certain threshold, then the module confirms the user's identity.

D. RIGHT-HAND DETECTION MODULE

The right-hand detection module (part of the skeleton detection) searches through the main subject's detected joints for the right hand's wrist. The right-hand detection module sends the right-hand region to the gesture classification module, see Figure 3. In Section IV, we evaluated the algorithm's behavior in various multi-subject environment scenarios, including hidden right hand and no gesture by the main subject.

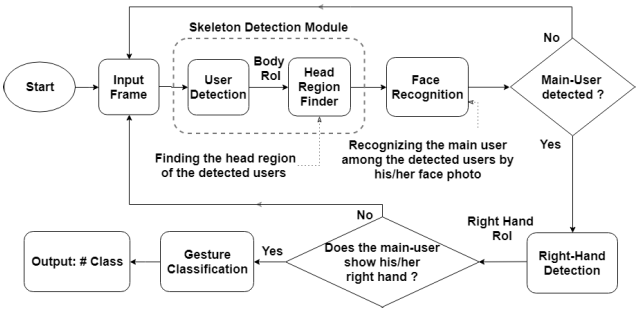

FIGURE 3. Flowchart of the hand gesture recognition in a multi-subject environment.

E. HAND GESTURE CLASSIFICATION

We apply the CNN model to classify the right-hand gesture into one of the four specified right-hand gestures. Each gesture represents a particular command that is given to the multi-rotor aerial vehicle as a control signal. The considered gesture classes are Palm gesture for take-off, Fist gesture for landing, Victory sign gesture for turning left, and OK sign gesture for turning right command.

To train the classifier, we use the Keras library [32]. The hand gesture classification has three main modules: (i) a classification module, (ii) a classification validation module, and (iii) a re-trainer module, Figure 4. The right-hand RoI and the classified gesture are the primary input and output of this unit, respectively. Also, an internal validation module checks the classifier's performance and activates the re-trainer in case the classifier performance was not satisfying.

1) RIGHT-HAND CLASSIFIER

Upon receiving the detected right-hand RoI as an input image, a CNN model classifies the detected RoI into one of the predefined classes. The classified output is then sent to the classification validation module to check the classifier performance. Furthermore, the classified output is mapped to one of the pre-determined commands. The command is then sent to the system validation module to monitor the system performance and to the next module for command execution, Figure 4.

2) RE-TRAINER

Re-trainer's module responsibility is to take S new images as input and train the current CNN model with the newly gathered images. The re-trainer module then sends the newly obtained weights to the CNN model to improve the classifier performance. The re-trainer module is triggered when either: (i) the classification validation module detects low confidence in the classifier performance; or (ii) the user

TABLE 1. Comparison between the proposed method in multi-subject environment and similar approaches.

Other Approaches / Criteria	Method [9]	Method [10]	Method [11]	Method [12]	Proposed Method
Number of Gestures	13 Continuous Commands	7 Continuous Commands	9 Continuous Commands	13 Continuous Commands	20 Continuous and Discrete Commands
Increasing the Gestures	✓	✗	✓	✗	✓
Classifier	P-CNN	No Classifier Just Filtering	CNN	MLP	Right Hand: SSD + CNN + R-CNN Left Hand: SVM
Classifier Accuracy	91.9%	82%	81.5%	92%	94.28%
Adaptability with New Users	✓	✗	✗	✗	✓
Real-Time Processing	✓	✓	✓	✓	✓
Output Validation	✗	✗	✗	✗	✓
Commands Sources	Camera	Camera	Wearable IMU and MMG Sensors	Wearable Accelerometer and Gyroscope Sensors	Camera and Wearable MPU and Flex Sensors on a Glove
Independent of the Surrounding Environment	✓	✗	✓	✓	✓

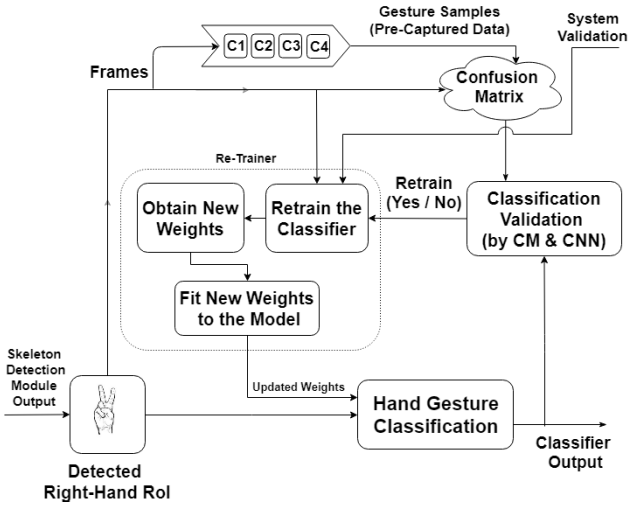

FIGURE 4. Gesture classification block diagram for a multi-subject environment.

declines the drone output with the smart glove, which then the system validation module triggers the re-trainer module, Figure 4. Retraining and validation process for single- and multi-subject environment cases are explained in Algorithm1 and Algorithm2, given in the Appendix section.

IV. EXPERIMENTAL RESULTS

The system performance is evaluated under various scenarios and penalties on the controller and the classifier. Here we demonstrate the system performance for the multi-subject environment using a LOGI HD 1080*p* webcam. For the algorithm implementation, an external Linux-based GPU, Nvidia Jetson AGX Xavier, was used. The GPU algorithm run-time for one and two subjects per frame is three frames per second (FPS), and for three and four subjects per frame is five to six FPS. Furthermore, Table 1 provides a comparison between the proposed method and similar methods based on specific criteria such as the number of gestures, classifier accuracy, validation modules, ability to increase the number of gestures, adaptability with a new user, type of classifiers, and the source of interaction between the user and the system.

As shown in Table 1, while none of the approaches has validations and the user must be alone in front of the camera, our proposed method has three validations, and supports cases with multiple subjects in front of the camera.

In order to evaluate the accuracy and study behavior of the proposed algorithm, a data-set of subjects showing different, pre-defined, right-hand gestures was created. We randomly selected n subjects with different physical features to take their pictures in different situations while the subjects are showing pre-defined gestures in front of the camera in various group sizes. We distributed n subjects in groups of m subjects as shown in Table 2. After filtering and excluding blurry and noisy images, we used the remaining images to test the proposed method. Furthermore, we added another class in case the main subject does not perform any gesture and keeps his/her right hand down. We tested our algorithm on different subjects based on the number of subjects in each picture. Table 2 shows the results of the experiment in each group.

A. VALIDATION MODULES

The proposed algorithm has three validation modules with a goal to improve the overall system performance. A user is the first to validate the system output by observing the drone's behavior and can send his/her disapproval through the smart glove by performing a dedicated gesture (*Gesture*#15). This specific gesture was chosen to code the observed misbehavior and trigger the system validation that monitors both outputs – the classifier output and the flight controller output signals to find the misbehavior source. Second, the system's outer layer has a validation module that identifies the misbehavior source, which can be either with the classifier or the flight controller. Finally, to validate the gesture classifier output, a classification validation module re-trains the classifier until the classifier confidence reaches the predefined threshold, which is set before the system starts [4].

B. SKELETON DETECTION

In order to detect all subjects in the input frame, we implemented a joint-based detection module [16] with an R-CNN

TABLE 2. Gesture recognition in a multi-subject environment experimental results for four different group populations.

Group Population	Number of Subjects	Total Images	Users Detection		Facial Recognition		Right-Hand Detection		Gesture Classification		Algorithm Accuracy %
			Correct	%	Correct	%	Correct	%	Correct	%	
1	35	140	140	100	140	100	140	100	129	92.14	98.04
2	20	760	756	99.47	755	99.87	755	100	679	89.93	97.32
3	16	4,480	4,395	98.1	4,309	98.04	4,258	98.82	3,601	87.57	94.88
4	10	5,040	4,809	95.42	4612	95.9	4471	96.94	3137	70.16	89.61

network on the GPU with 15 FPS. The proposed algorithm maps the joints from top to bottom.

C. FACIAL RECOGNITION

For the facial recognition model, we used a condition-free facial recognition library (no illumination changes or occlusions) with 99.38% accuracy that needs just one sample image for training [17]. The facial recognition model creates the landmark based on the provided single face image from the main subject and uses it to detect the main subject's face among the detected head RoIs from the skeleton detection module. A picture of the main subject's face is provided during the initialization phase.

D. HAND GESTURE RECOGNITION USING THE WEBCAM

To classify the right-hand gesture, the CNN model is used. Our data set to train this CNN model consists of 8125 images, 640 × 480 pixels, were collected from 65 subjects standing at various distances from the camera. Each subject stood in front of a white background with their right hand showing four pre-determined static gestures. After manually refining the images, a data set of 7600 JPEG pictures was obtained. We applied 70% of refined images (5320 images) to train the CNN model and used the rest (2280 images) as a test data set.

A sign of an overfitted model is when the training loss is low, and the validation loss is high [18]. As shown in Figure 5(b), the training loss is higher than the validation loss. Moreover, since its inception, both validation and training losses have diminished until they reached their lowest value after 14 epochs. In order to avoid the overfitting at each epoch, the data augmentation technique is used in the training process, Figure 5. The model was trained for 14 epochs with an accuracy of 95.83%. The batch size is 32, 'Adam' is chosen as the optimizer, and mean squared error is used as a loss function in the training process. After training, the system's accuracy of 300 images from each class (total of 1200 pictures that the system has not seen before) was 96.41%.

In the algorithm's implementation, four right-hand gestures were used; however, the proposed algorithm can handle different numbers of gestures, and the only part that requires adjustment is the dense of the last layer in the CNN model.

Table 3 shows the accuracy, loss values, and their related training data sizes. The proposed CNN model shows convergence for data size larger than 700 images for each class

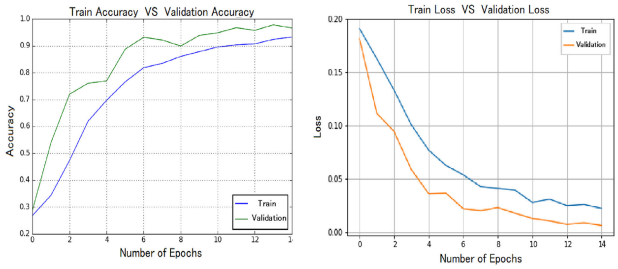

FIGURE 5. Training set versus validation set: (a) accuracy, (b) loss.

(gesture). The data augmentation technique based on vertical flipping and setting degree range for random rotation is used to reduce the required data size and reach higher accuracy.

TABLE 3. Right-hand gesture classifier training and validation results for different data sizes.

Data Size (number of images for each of four classes)	Training Accuracy %	Validation Accuracy %	Train Loss	Validation Loss
25	36.11	31.25	0.186	0.189
50	50	46.77	0.159	0.151
100	62.41	69.67	0.129	0.101
200	82.03	81.4	0.074	0.068
350	84.73	91.23	0.055	0.035
500	89.59	95.18	0.039	0.019
700	**94.29**	**96.8**	**0.023**	**0.01**
1150	94.36	96.94	0.021	0.012

To test the proposed method in a multi-subject environment, we randomly selected 35 single-subjects, 190 two-subject groups, 560 three-subject groups, and 210 four-subject groups. After manually filtering the blurry and noisy images, a dataset of 10420 JPEG color pictures was obtained. Each group was used to test all the main modules, such as skeleton detection, facial recognition, main subject's right-hand detection, and right-hand gesture classification.

Table 2 shows the results of single-, two-, three-, and four-subject groups separately. All of the 10420 captured images from the 81 subjects were used to test the multi-subject environment algorithm. The captured dataset of 10420 images is divided into 140 images of single-subject groups, 760 images of two-subject groups, 4480 images of three-subject groups, and 5040 images of four-subject groups. The final accuracy for the multi-subject environment is 94.96%.

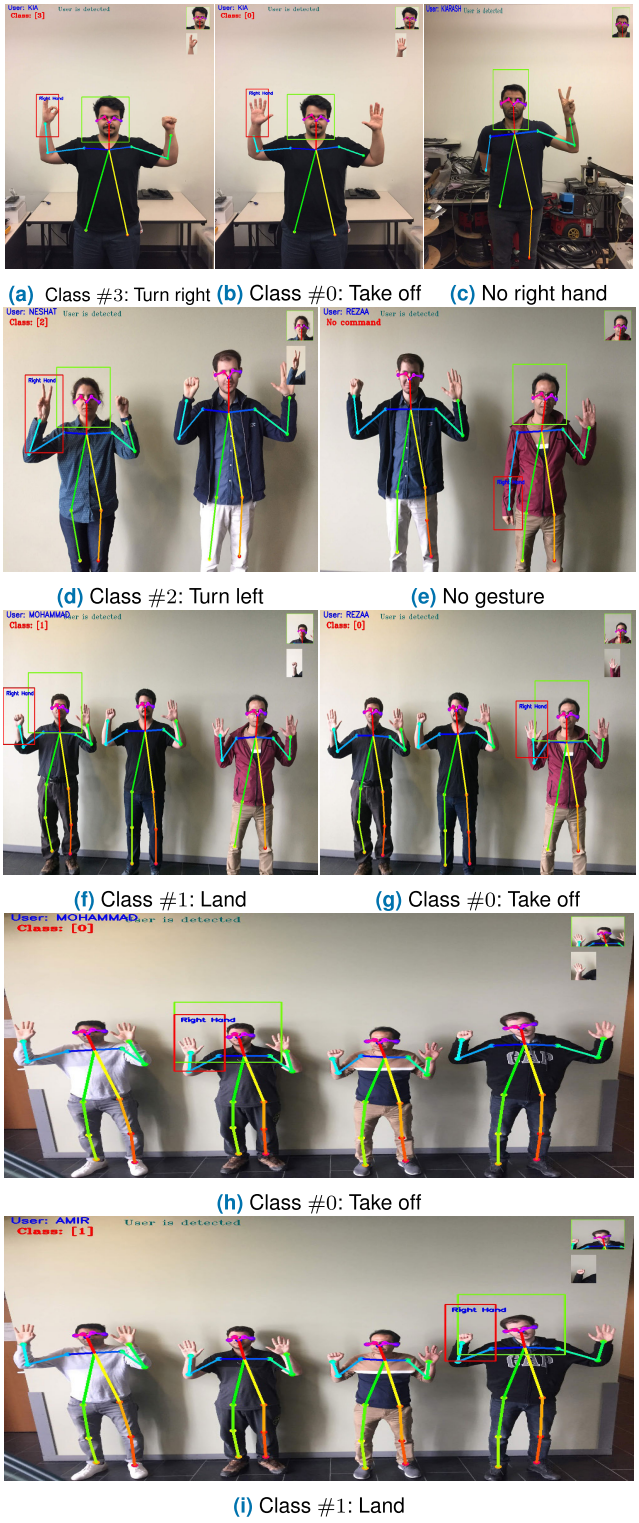

FIGURE 6. Gesture recognition on different groups of subjects including a main subject performing in front of the camera. The commands are: class #0: take off; class #1: land; class #2: turn left; and class #3: turn right.

E. CASE ONE: FRAME CONTAINS BOTH HANDS

In this scenario (Figure 6(a), Figure 6(b), Figure 6(d), Figure 6(f), Figure 6(g), Figure 6(h), and Figure 6(i)) the user shows both hands in proper angles such that the camera can capture one of the gestures. The algorithm detects the correct hand of the main subject and then sends the right-hand RoI to the classification module.

FIGURE 7. The experimented case: penalty on the classifier. The commands are: class #0: Take-off, class #1: Land, class #2: Turn left, and class #3: Turn right.

TABLE 4. Penalty on the classifier: Class #3. Classification validation decides to re-train the model on Class #3 in three steps, thus raising the confidence from 54.55% to 79.19%. The four classes are Class #0: take off, Class #1: land, Class #2: turn left, and Class #3: turn right.

Steps	Step #1				Step #2				Step #3			
Class	0	1	2	3	0	1	2	3	0	1	2	3
0	8	3	0	0	11	0	0	0	11	0	0	0
1	0	6	0	0	0	6	0	0	0	5	1	0
2	0	1	10	0	1	0	10	0	0	0	11	0
3	0	5	0	6	5	0	0	11	5	0	0	16
Conf %	72.73	100	90.91	54.55	100	100	90.91	68.73	100	83.33	100	79.19

F. CASE TWO: FRAME WITHOUT RIGHT HAND

In this scenario, Figure 6(c), the user shows a gesture with his left hand while his right hand is hidden. As the first step, the skeleton detection module detects the subject in the frame, sends his head RoI to the facial recognition module, and returns the confirmation that the main subject has been detected. Then, the skeleton detection module searches for the joint number of 4, and when it can not be found in the list of the detected joints, the process stops there until the user shows his right hand.

Furthermore, we examine the algorithm performance in three undesired scenarios: 1) the user holds right hand towards the ground; 2) the user shows both hands; 3) the user does not show any gesture and hides his/her hand, or the camera does not capture the correct hand.

G. CASE THREE: FRAME CONTAINS RIGHT HAND WITHOUT GESTURE

Here the user does not show any gesture and keeps his right hand down, Figure 6(e). The proposed algorithm detects the right-hand joint and is ready to capture the hand region. The algorithm calculates the shoulder joint and wrist joint angle before the algorithm sends the detected right-hand region to the classification module. The expected angle between the shoulder and wrist is between 0 to 90 degrees. If a user opens the right hand more than this range, the algorithm assumes that the right-hand points towards the ground and the result is "No Gesture."

H. CASE FOUR: PENALTY ON THE CLASSIFIER

In this case, the user confronts the wrong behavior from the drone. The user shows the "Land" command, which is *Gesture*#1, but the drone turns right instead of landing, see Figure 7. Hence, the user declines the drone action with the smart glove and creates *Gesture*#15. The system validation module discovers that the controller output meets the

Algorithm 1: Pseudo-Code for the Retraining and Validation Process in a Multi-Subject Environment

```
begin
    Subjects pose in front of the camera;
    Skeleton detection module ← Input frame;
    Facial recognition module ← Heads RoI;
    Skeleton detection module ← Main subject ID;
    Classifier ← Hand RoI;
    Flight controller ← Classification unit output;
    Drone ← Flight controller output parameters;
    User observes the drone behavior;
    if User disapproves the drone behavior then
        User sends a command to the system validation
          module through the smart glove;
        System validation module checks the flight
          controller output changes;
        if The changes of the flight controller output
          were not as expected then
            System validation module adjusts the flight
              controller;
        else
            System validation modules activates the
              classifier re-trainer;
            while Gestures confidence values are less
              than predefined threshold do
                The re-trainer captures S new images
                  from the low confidence gesture;
                Re-trainer re-trains the running CNN
                  model with the newly gathered data in
                  order to obtain new weights;
            Re-trainer fits the new weights to the CNN
              model, and sends the CNN model to the
              classifier;
```

Algorithm 2: Pseudo-Code for the Validation Process in a Single-Subject Environment

```
begin
    User poses in front of the camera;
    Hand detector ← Input frame;
    Classifier ← Hand RoI;
    Flight controller ← Classification unit output;
    Drone ← Flight controller output parameters;
    User observes the drone behavior;
    if User disapproves the drone behavior then
        User sends a command to the system validation
          module through the smart glove;
        System validation module checks the flight
          controller output parameters changes;
        if The changes of the flight controller output
          parameter were not as expected then
            System validation module adjusts the flight
              controller;
        else
            System validation modules activates the
              classifier re-trainer;
            while Confidences of the gestures are less
              than predefined threshold do
                The re-trainer captures S new images
                  from the low confidence gesture;
                Re-trainer re-trains the running CNN
                  model with the newly gathered data in
                  order to obtain new weights;
            Re-trainer fits the new weights to the CNN
              model, and sends the CNN model to the
              classifier;
```

classifier output and concludes that the problem is with the classifier. Subsequently, the system validation module transfers the discipline command to the re-trainer in the classification module.

As shown in Table 4, when the classifier re-trainer is activated, S images (in our implementation $S = 5$) of the gesture with the lowest confidence level are taken by sending a specific command, i.e., *Gesture*#15, using the smart glove. After gathering S images with the described system configuration, it takes 4.2 seconds to re-train the classifier and fit new weights into the running CNN model. After each step (gathering images, re-training based on gathered images, and fitting the new weights to the running CNN model), if the confidence is less than the threshold (determined by the user before the system starts; in this experiment set to 70%), the classification validation module reactivates the re-trainer. Repeatedly, the re-trainer captures the images by the user's command, re-trains based on those images, and fits the CNN model. As shown in Table 4, the classification validation module triggers the re-trainer until all the calculated confidence values are above the threshold.

V. CONCLUSION

We presented a gesture-based control system for multi-rotor aerial vehicles that detects and classifies both right- and left-hand gestures in a multi-subject environment. There are no limitations such as requiring a user to stand alone in front of the camera or show just one hand. The proposed algorithm distinguishes between subjects to find the main subject and classifies the user's right-hand gesture.

The proposed system operates with two different human-robot interaction subsystems: (i) the smart glove that produces continuous and discrete control commands, and (ii) an image processing method that produces discrete commands using hand tracking and gesture recognition. The proposed system uses separate modules to find the correct hand's region in the input frame before the hand gesture recognition module recognizes the gesture, which improves the gesture recognition accuracy. An online learning method is developed to improve system reliability and allow adaptation to new users.

The proposed method contains three separate validation modules to improve system performance. The proposed method's performance was evaluated for cases including penalties on the classifier, undesired user performance, and subjects with various physical features.

Future work will include developing a gesture-based control system that can detect multiple main subjects in a multi-subject environment and classify their hand gestures. Furthermore, an inertial measurement unit (IMU) sensor will be embedded in the smart glove to measure the hand's acceleration and rotation. Additionally, an electromyography (EMG) sensor will be used on the main subject's arm to gather electrical signals generated by the main subject's muscles. We will use these newly gathered data to train an SVM model to classify additional gestures and create additional commands for control purposes.

APPENDIX A: PSEUDO-CODE FOR THE RETRAINING AND VALIDATION PROCESS

Here we present two pseudo-codes of algorithms used in the validation process in single- and multi-subject environments.

REFERENCES

[1] T. Chouhan, A. Panse, A. K. Voona, and S. M. Sameer, "Smart glove with gesture recognition ability for the hearing and speech impaired," in *Proc. IEEE Global Humanitarian Technol. Conf. South Asia Satell. (GHTC-SAS)*, Sep. 2014, pp. 105–110.

[2] M. Obaid, O. Mubin, C. A. Basedow, A. A. Ünlüer, M. J. Bergström, and M. Fjeld, "A drone agent to support a clean environment," in *Proc. 3rd Int. Conf. Human-Agent Interact.*, 2015, pp. 55–61.

[3] Z. Lu, X. Chen, Q. Li, X. Zhang, and P. Zhou, "A hand gesture recognition framework and wearable gesture-based interaction prototype for mobile devices," *IEEE Trans. Human-Mach. Syst.*, vol. 44, no. 2, pp. 293–299, Apr. 2014.

[4] K. Haratiannejadi, N. E. Fard, and R. R. Selmic, "Smart glove and hand gesture-based control interface for multi-rotor aerial vehicles," in *Proc. IEEE Int. Conf. Syst., Man Cybern. (SMC)*, Oct. 2019, pp. 1956–1962.

[5] K. K. Lekkala and V. K. Mittal, "Accurate and augmented navigation for quadcopter based on multi-sensor fusion," in *Proc. IEEE Annu. India Conf. (INDICON)*, Dec. 2016, pp. 1–6.

[6] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[7] A. Sanna, F. Lamberti, G. Paravati, and F. Manuri, "A kinect-based natural interface for quadrotor control," *Entertainment Comput.*, vol. 4, no. 3, pp. 179–186, Aug. 2013.

[8] S. Gaglio, G. L. Re, and M. Morana, "Human activity recognition process using 3-D posture data," *IEEE Trans. Human-Mach. Syst.*, vol. 45, no. 5, pp. 586–597, Oct. 2015.

[9] A. G. Perera, Y. Wei Law, and J. Chahl, "UAV-GESTURE: A dataset for UAV control and gesture recognition," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018.

[10] N. SaiChinmayi, C. Hasitha, B. Sravya, and V. K. Mittal, "Gesture signals processing for a silent spybot," in *Proc. 2nd Int. Conf. Signal Process. Integr. Netw. (SPIN)*, Feb. 2015, pp. 756–761.

[11] Y. Ma, Y. Liu, R. Jin, X. Yuan, R. Sekha, S. Wilson, and R. Vaidyanathan, "Hand gesture recognition with convolutional neural networks for the multimodal UAV control," in *Proc. Workshop Res., Edu. Develop. Unmanned Aerial Syst. (RED-UAS)*, Oct. 2017, pp. 198–203.

[12] S.-Y. Shin, Y.-W. Kang, and Y.-G. Kim, "Hand gesture-based wearable human-drone interface for intuitive movement control," in *Proc. IEEE Int. Conf. Consum. Electron. (ICCE)*, Jan. 2019, pp. 1–6.

[13] A. S. M. M. Rahman, J. Saboune, and A. El Saddik, "Motion-path based in car gesture control of the multimedia devices," in *Proc. 1st ACM Int. Symp. Design Anal. Intell. Veh. Netw. Appl.*, 2011, pp. 69–76.

[14] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, and A. Blake, "Efficient human pose estimation from single depth images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2821–2840, Dec. 2013.

[15] Y. Wang, A. Ren, M. Zhou, W. Wang, and X. Yang, "A novel detection and recognition method for continuous hand gesture using FMCW radar," *IEEE Access*, vol. 8, pp. 167264–167275, 2020.

[16] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," 2018, *arXiv:1812.08008*. [Online]. Available: http://arxiv.org/abs/1812.08008

[17] P. Jothi Thilaga, B. Arshath Khan, A. A. Jones, and N. Krishna Kumar, "Modern face recognition with deep learning," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, Apr. 2018, pp. 1947–1951.

[18] S. Park and N. Kwak, "Analysis on the dropout effect in convolutional neural networks," in *Proc. Asian Conf. Comput. Vis.* Springer, 2016, pp. 189–204.

[19] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," in *Proc. Int. Conf. Rough Sets Knowl. Technol.* Springer, 2014, pp. 364–375.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, Jun. 2014.

[21] S. Cai, Y. Shu, G. Chen, B. Chin Ooi, W. Wang, and M. Zhang, "Effective and efficient dropout for deep convolutional neural networks," 2019, *arXiv:1904.03392*. [Online]. Available: http://arxiv.org/abs/1904.03392

[22] B. Karlik and A. V. Olgac, "Performance analysis of various activation functions in generalized mlp architectures of neural networks," *Int. J. Artif. Intell. Expert Syst.*, vol. 1, no. 4, pp. 111–122, 2011.

[23] Z. Zhang, "Improved Adam optimizer for deep neural networks," in *Proc. IEEE/ACM 26th Int. Symp. Quality Service (IWQoS)*, Jun. 2018, pp. 1–2.

[24] E. M. Dogo, O. J. Afolabi, N. I. Nwulu, B. Twala, and C. O. Aigbavboa, "A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks," in *Proc. Int. Conf. Comput. Techn., Electron. Mech. Syst. (CTEMS)*, Dec. 2018, pp. 92–99.

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[26] L. Santos, N. Carbonaro, A. Tognetti, J. L. González, E. De la Fuente, J. C. Fraile, and J. Pérez-Turiel, "Dynamic gesture recognition using a smart glove in hand-assisted laparoscopic surgery," *Technologies*, vol. 6, no. 1, p. 8, 2018.

[27] R. Voyles, J. Bae, and R. Godzdanker, "The gestural joystick and the efficacy of the path tortuosity metric for human/robot interaction," in *Proc. 8th Workshop Perform. Metrics Intell. Syst.*, 2008, pp. 91–97.

[28] A. S. Al-Shamayleh, R. Ahmad, M. A. M. Abushariah, K. A. Alam, and N. Jomhari, "A systematic literature review on vision based gesture recognition techniques," *Multimedia Tools Appl.*, vol. 77, no. 21, pp. 28121–28184, Nov. 2018.

[29] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: A survey," *Artif. Intell. Rev.*, vol. 43, no. 1, pp. 1–54, Jan. 2015.

[30] H. S. Hasan and S. A. Kareem, "Human computer interaction for vision based hand gesture recognition: A survey," in *Proc. Int. Conf. Adv. Comput. Sci. Appl. Technol. (ACSAT)*, Nov. 2012, pp. 55–60.

[31] O. Déniz, G. Bueno, J. Salido, and F. De la Torre, "Face recognition using histograms of oriented gradients," *Pattern Recognit. Lett.*, vol. 32, no. 12, pp. 1598–1603, Sep. 2011.

[32] F. Chollet, "Keras: The python deep learning library," ASCL, Tech. Rep. ASCL-1806, 2018.

KIANOUSH HARATIANNEJADI received the B.S. degree in electrical engineering from the Hamadan University of Technology, Hamadan, Iran, in 2016. He is currently pursuing the M.A.Sc. degree in electrical and computer engineering with Concordia University, Montreal, QC, Canada. His research interests include machine learning, gesture detection and classification, computer vision, and hand movement-based control.

RASTKO R. SELMIC (Senior Member, IEEE) received the B.S. degree in electrical engineering from the University of Belgrade, Belgrade, Serbia, in 1994, and the M.S. and Ph.D. degrees in electrical engineering from The University of Texas at Arlington, Arlington, TX, USA, in 1997 and 2000, respectively.

From 1997 to 2002, he was a Lead DSP Systems Engineer with Signalogic Inc., Dallas, TX, USA. He joined the Electrical Engineering Department, Louisiana Tech University, Ruston, LA, USA, in 2002, where he was an AT&T Professor of electrical engineering, from 2014 to 2016. He is currently a Professor and an Associate Chair of Graduate Studies with the Department of Electrical and Computer Engineering, Concordia University, Montreal, QC, Canada. He consulted for numerous companies, including Andrew Corporation, Richardson, TX, USA, Intelligent Automation Inc., Rockville, MD, USA, Davis Technologies International Inc., Dallas, TX, USA, and American GNC Corporation, Simi Valley, CA, USA. He has authored/coauthored 100 journal and conference papers, four book chapters, and books *Wireless Sensor Networks: Security, Coverage, and Localization* (Springer 2016) and *Neuro-Fuzzy Control of Industrial Systems with Actuator Nonlinearities* (SIAM Press, Philadelphia, PA, 2002). He holds one U.S. patent with four additional reports of invention. His current research interests include smart sensors and actuators, cooperative sensing and control, gesture-based computing and control, and UAV control.

Dr. Selmic served as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON CYBERNETICS.