Sensor Dataglove for Real-time Static and Dynamic Hand Gesture Recognition

Md Ahasan Atick Faisal  
Department of Electrical and Electronic Engineering  
University of Dhaka  
Dhaka-1000, Bangladesh  
Email: atickfaisal@gmail.com

Farhan Fuad Abir  
Department of Electrical and Electronic Engineering  
University of Dhaka  
Dhaka-1000, Bangladesh  
Email: farhanfuad.abir@gmail.com

Mosabber Uddin Ahmed  
Department of Electrical and Electronic Engineering  
University of Dhaka  
Dhaka-1000, Bangladesh  
Email: mosabber.ahmed@du.ac.bd

Abstract—Hand gesture recognition has been a widely explored field of Human Activity Recognition (HAR). In this work, we have presented a sensor-based hand gesture recognition framework to classify both static and dynamic hand gestures in real-time using a dataglove that contains a 3-axis accelerometer (ACC), a 3-axis gyroscope, and 5 flex sensors. We have collected data from 35 volunteers performing 14 static and 3 dynamic gestures wearing the dataglove. We have preprocessed the raw flex sensor data using digital filtering techniques and performed mathematical operations on the accelerometer and gyroscope data for determining accurate orientation and motion profile. Four classical machine learning algorithms were used and compared on both datasets. We have achieved maximum accuracy of 99.53% for static gestures and 98.64% for dynamic gestures using the K-Nearest Neighbors (KNN) classifier. Our proposed framework provides real-time wireless hand gesture detection for Human-Computer Interaction (HCI) and Sign Language Recognition (SLR).

Contribution—We developed a multimodal hardware and software framework for recognizing 17 hand gestures in real-time.

Keywords—Human-computer interaction; real-time hand gesture recognition; dataglove; flex sensor; IMU; KNN

I. INTRODUCTION

With the recent development in processing capability and networking, we are experiencing a paradigm shift in pervasive computing-based researches. Though this trend of technology development has been around since the ‘90s, the core approaches have been explored more extensively in the last few years. The primary aim of these researches is to make the interaction between humans and computers more natural and spontaneous. As hand gesture is one of the primary mediums of natural communication among humans, interpretation of human hand gesture by computer is also one of the primary fields of Human-Computer Interaction (HCI).

The first step of any HCI-based system is to analyze and quantify human movements to distinguish the meaningful signs. Hand gesture is part of the macro-movement subcategory of human body movement taxonomy [1]. If these types of macro-movements can be interpreted by the computer, then interaction with the computer can be more natural. Recently, another practice in technology development has been the Internet of Things (IoT) which has enabled widespread connectivity among different devices. This opened new application domains for hand gesture recognition systems. Moreover, the increase in computational capability, as well as computationally efficient algorithms have also opened new device-level machine learning scopes. As a result, like other fields of HCI, new possibilities in the field of hand gesture recognition are explored to date.

Hand gesture recognition modalities can be divided into two main categories: computer vision-based approaches and sensor-based approaches. Computer vision-based approaches include recording and analysis of gesture images. On the other hand, sensor-based approaches include devices containing different sensors to sense the change in signals during gesture performance. Both approaches have their merits and demerits. The vision-based approaches can perform gesture recognition for more than one participant in the field of view but it is highly dependent on different environmental factors like illumination, occlusion, and position of the image capturing device [2], [3]. On the other hand, the wearable sensors are easy to implement, generally are not affected by environmental dynamics rather provide more reliable gesture profiles [4].

The invention of sensor-based hand gesture recognition devices goes way back to the last century during the very first researches in HCI. The first commercial dataglove was developed by VPL Inc. as a VR product in 1987 [5]. The researchers behind this invention presented two versions of this dataglove - DataGlove and Z-Glove. They invented the optical flex sensor for sensing finger movements. The first one was for Apple Macintosh and had a magnetic positioning and orientation tracking system while the Z-Glove was developed for Commodore 64 and the positioning and orientation tracking system was ultrasound-based [6]. However, these types of datagloves were very costly at that time. Moreover, different companies, racing for cutting-edge technologies, implemented and developed more such devices which enabled users to interact with the virtual environment with ease. The development of miniature IMU sensors enabled low-cost dataglove development and our work is based on such a sensor.

Nowadays, deep learning-based classifiers are outperforming nearly all classical machine learning classifiers. But these
systems need greater computational capability. This becomes more crucial for embedded systems implementations. In most recent studies, the hand gesture detecting devices are not portable. On the other hand, real-time detection systems need less complex classifier models to minimize the run-time. So, this work has aimed to balance the algorithm complexity, run-time, and portability and develop a sensor-based user-independent real-time gesture recognition system with wireless connectivity.

The rest of the paper has been arranged as follows: Section 2 summarizes recent developments and the previous work related to hand gesture recognition systems. Section 3 discusses the methodology of our framework in detail. Section 4 presents the result of our research. Finally, we have drawn conclusion to this work in Section 5.

II. RELATED WORKS

Over the year, sensor-based hand gesture recognition modalities have evolved from datagloves to the state of the art contactless approaches. Notable recent hand gesture recognition works using these modalities are as follows.

Different sensors detect different gesture features. That is why one sensor may perform better in certain scenarios than others. The choice of the sensor depends mostly on the application of the framework. A detailed analysis of the Surface Electromyography (sEMG) sensor and accelerometer sensor shows that fusion of both sensors increase the classification accuracy of single finger movement, multi-finger movement, and wrist movement [7]. So, the fusion of sensors is preferred in most cases. Wii Remote (Wiimote), an accelerometer-based gaming remote, was developed based on Hidden Markov Model (HMM) and Bayes classifier [8] and was capable of classifying Square, Circle, Roll, Z and Tennis. SoapBox is a hand-held module containing accelerometer sensors that were used to measure the dynamic acceleration and tilt of the device [9]. A Discrete HMM was used for the classification of hand gestures.

In Sign Language Recognition (SLR), datagloves have been used in different researches. A triaxial accelerometer and multi-channel electromyography (EMG) sensor-based SLR framework was proposed which could classify 72 Chinese Sign Language gestures [10]. A flex-sensor and Inertial Measurement Unit (IMU) sensor which contains 3-axis accelerometer and 3-axis gyroscope in the same package. For sensor data processing and connectivity, we used DOIT Esp32 DevKit v1 board.

B. Software Framework

The dataglove is connected to the central server via MQTT protocol. From the central server, our developed software is
capable of recording data during the data collection stage. According to Figure 2b, the bar graph shows the normalized flex data and the spider plot shows the yaw, pitch, and roll angles. Later, during the detection stage, the software provides gesture recognition output in real-time.

C. Data Collection

We gathered hand gesture data from 35 volunteers (25 male and 10 female) in total. They were in the age groups of 21-32 years. We recorded static gesture from 30 volunteers (21 male and 9 female) and dynamic gestures from the rest. Our 14 selected static hand gestures are come here, go away, fist, fingers crossed, cash, one, two, three, four, five, excellent, stop, thumbs up and thumbs down. Each volunteer performed the static gestures with 10 time repetition. Moreover, we included 3 dynamic hand gestures: painting, sorry and thank you. These gestures are showed in Figure 3. Each gesture has different hand movement and orientation. The dataglove recorded sensor values with sampling frequency of 7 Hz.

D. Preprocessing

The analog sensor data from the flex sensors often contain noise. The orientation and motion determination processes can be performed in different ways. The data filtering and IMU data processing steps are described below:

a) Moving Average Filter: The analog flex sensor data generally contains small changes around a certain value for any orientation. So we passed the raw analog data through a moving average filter to reduce the noise. The filter equation is as follows:

\[ s[n] = \frac{1}{N} \sum_{k=0}^{N-1} r[n-k] \]  

where \( s[n] \) denotes the filtered output for \( n \)th sample, \( N \) is the number of samples to perform moving average on and \( r[n] \) is raw data for \( n \)th sample. The higher value of \( N \) gives a very smooth output but adds reaction delay. For our framework, we performed moving average on 5 consecutive values which resulted in a relatively smooth signal with fast enough response for real-time application.

b) Digital Motion Processor: The data from the on-chip Digital Motion Processor (DMP) of MPU-6050 contains 10 blocks in total - 4 blocks for quaternions \( (q_0, q_1, q_2, q_3) \) shown in equation 2), 3 blocks for accelerometer data \( (acc_x, acc_y, acc_z) \) and 3 blocks for gyroscope data \( (gyr_x, gyr_y, gyr_z) \).

c) Calculation of Orientation Angles: The DMP of MPU-6050 outputs the orientation as quaternion values. A general form of quaternion is given by the following expression [21]:

\[ Q = q_0 + q_1i + q_2j + q_3k \]  

where quaternion \( Q \) is the addition of a scaler \( q_0 \) and a vector \( q = (q_1, q_2, q_3) \).

The euler angles can be calculated from the quaternion values by the following equation:

\[
\begin{bmatrix}
\phi \\
\theta \\
\psi
\end{bmatrix} =
\begin{bmatrix}
\tan^{-1}\left(\frac{2(q_0q_1+q_2q_3)}{1-2(q_1^2+q_2^2)}\right) \\
\sin^{-1}\left(\frac{2(q_0q_2-q_1q_3)}{\sqrt{1-2(q_1^2+q_2^2)}}\right) \\
\tan^{-1}\left(\frac{2(q_0q_3+q_1q_2)}{1-2(q_2^2+q_3^2)}\right)
\end{bmatrix}
\]  

where \( \phi, \theta \) and \( \psi \) are roll, pitch and yaw angles respectively. These Euler angles are used to express device orientation.

d) Calculation of Linear Acceleration: The DMP provides raw accelerometer data which contains the device acceleration including the gravity. So, there is always a gravity offset in the acceleration value. In order to derive the actual acceleration of the device, we have calculated and subtracted the gravity contributions from three-axis acceleration values. This formula for linear acceleration calculation is given below [22]:

\[
\text{accel} = \text{acc}_{raw} - \text{acc}_{gravity}
\]
Dynamic Gesture

1.43 s

50%

that it can produce a real-time gesture profile.

appropriate classification algorithms afterward. In case of real-

time operation, window length is needed to be defined such

are extracted from each window samples which are fed to the

used. To perform data segmentation, at first, sensor signals are

divided into smaller time segments or windows. Then features

of time-series data, different segmentation methods can be

For dynamic gestures, in order to extract relevant properties

to North-East-Down (NED) coordinate system [23]. The

linear acceleration value is rotated from body coordinate sys-

In order to remove the orientation dependency, the raw lin-

accelerations in three axes.

∥LACC∥

∥g∥

acc

1 − 2(q_2^2 + q_3^2)

2(q_1q_2 - q_0q_3)

2(q_1q_3 + q_0q_2)

lacc_x

acc_x

acc_y

acc_z

LACC_x

LACC_y

LACC_z

where \( q_0, q_1, q_2 \) and \( q_3 \) represent the quaternion values. \( LACC \) and \( lacc \) are respectively rotated and raw linear accelerations in three axes.

E. Data Segmentation

For dynamic gestures, in order to extract relevant properties

time-series data, different segmentation methods can be

are divided into smaller time segments or windows. Then features

are extracted from each window samples which are fed to the

appropriate classification algorithms afterward. In case of real-

time operation, window length is needed to be defined such

that it can produce a real-time gesture profile.

There is a trade-off between accuracy and reaction delay

for window length. The longer the window size, the better a

 certain gesture can be expressed by that window but it causes

reaction delay. So, an optimum window length is chosen

based on the duration of a single gesture [25]. The window

specifications of our system are shown in Table I.

| Number of samples per window | 10 |
| Window Length | 143 s |
| Window overlapping | 50% |

F. Feature Extraction

The hardware framework collects in total 14 channels of

sensor data. Flex sensors provide 5 channels, 3-axis rotated

linear acceleration, 3-axis gyroscope and orientation angles

(yaw, pitch, and roll) provide 3 channels each. But we have

not used all channels for both static and dynamic gestures. We

analyzed the hand motion taxonomy and based on the finger

flex angles and hand orientation variation, we classified the

static gestures for no variation in flex angles and the dynamic

gestures for any variation in the hand position or orientation

[26].

For static gestures, due to stationary hand position, there

is no data variation with time. So, data segmentation was not

performed in this case. Five flex sensors and 3 orientation

angles were used as features for static gesture recognition.

For dynamic hand gestures, before performing feature ex-

traction, each feature was scaled between \(-1\) to \(1\) beforehand

by the following equation:

\[ X = \frac{x - \mu_x}{\text{range}(x)} \] (8)

where \( X \) is the normalized feature, \( x \) is the features before

scaling, \( \mu_x \) denotes the mean, and \( \text{range}(x) \) represents the
difference between maximum and minimum value of that fea-
ture. Statistical features like mean, median, standard deviation,
Fig. 4: Confusion matrices for static and dynamic gestures

and root mean square (RMS) were calculated for each window of each channel. Afterward, each feature was closely inspected for each dynamic gesture, and from the inspection, we have found that the difference of standard deviation between x-axis acceleration and z-axis acceleration \((\text{std}(x) - \text{std}(z))\) show distinct waveforms and also provide the best classification accuracy for the dynamic gestures we used.

G. Classification

We trained four different classifiers: K-Nearest Neighbors (KNN), Random Forests (RnF), Support Vector Machine (SVM), and Decision Tree (DT) for both classification tasks. All classifiers have produced fair prediction results. Afterward, we tuned the parameters of each classifier that showed performance improvement. To optimize the parameters of our model, we performed a grid search with cross-validation. The training process was done on a desktop computer with the following configuration.

- Processor - Intel Core i5-5200U CPU @ 2.20GHz × 4
- GPU - Mesa Intel® HD Graphics 5500 (BDW GT2)
- Memory - 8 GB

<table>
<thead>
<tr>
<th>Static Gestures</th>
<th>Dynamic Gestures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier</td>
<td>Accuracy</td>
</tr>
<tr>
<td>KNN (K = 3)</td>
<td>99.53%</td>
</tr>
<tr>
<td>Decision Tree</td>
<td>98.74%</td>
</tr>
<tr>
<td>RnF</td>
<td>97.48%</td>
</tr>
<tr>
<td>SVM (RBF)</td>
<td>94.97%</td>
</tr>
</tbody>
</table>

IV. EXPERIMENTAL RESULTS AND DISCUSSION

The datasets for both classification tasks nearly have equal distribution. So, we chose ‘accuracy’ as the metric. The overall accuracy of the four models for both tasks are shown in Table II.

We have achieved the best accuracy with KNN for both static and dynamic gestures. The confusion matrices of KNN for static and classification tasks are shown in Figure 4. From the above evaluation methods, we can conclude that all these four classifiers have shown over 90% classification for both static and dynamic gestures. KNN classifiers have shown the best accuracy in both gesture types.

a) Misclassifications in Static Gestures: The system makes a few wrong predictions for ‘Come Here’ and ‘Five’. We have observed that during performing these two gestures, the hand orientations were close for a few volunteers. Moreover, the only difference between the sensor signals for ‘Stop’ and ‘Five’ has been the yaw angle. These resulted in a few wrong predictions. For static gestures, the SVM classifier faces difficulties classifying ‘Come Here’, ‘Five’, ‘Fist’, and ‘Stop’ gestures and RnF classifier faces difficulties classifying the ‘Come Here’ gesture. Decision Tree shows quite a good accuracy and does not face any crucial complication for any particular gesture. KNN (K=3) classifier shows the best overall classification accuracy of 99.53%. So, we can conclude that the KNN classifier is preferred for practical gesture recognition applications using this method.

b) Misclassifications in Dynamic Gestures: Among dynamic gestures, we saw wrong predictions on a few occasions for ‘Sorry’ and ‘Painting’ gestures. The probable reason for such error is that we have only used the rotated linear acceleration data for dynamic gesture classification. By adding gyroscope sensor data and flex sensor data, the result may improve. But according to the evaluation processes we have found decent accuracy rate, and the speed of recognition is nearly real-time.
TABLE III: Comparison of similar sensor-based hand gesture recognition systems

<table>
<thead>
<tr>
<th>Author</th>
<th>Number of Gestures</th>
<th>Type of Gestures</th>
<th>Sensor</th>
<th>Extracted Features</th>
<th>Classifier</th>
<th>Wireless Connectivity</th>
<th>Number of Participants</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chen et al. [7]</td>
<td>24</td>
<td>Static, Dynamic [Selected ASL+CSL, Control Command]</td>
<td>Accelerometer, sEMG</td>
<td>Mean Absolute Values (MAV), Ratio of MAVs, 4th-order AR model coefficients</td>
<td>Linear Bayesian Classifier</td>
<td>N/A</td>
<td>5</td>
<td>98%</td>
</tr>
<tr>
<td>Zhang et al. [10]</td>
<td>110</td>
<td>Static, Dynamic [Interaction-oriented, CSL]</td>
<td>Accelerometer, sEMG</td>
<td>Mean, Std, MAV, 4th-order AR coefficients</td>
<td>Decision Tree, HMM</td>
<td>N/A</td>
<td>2</td>
<td>90.2%</td>
</tr>
<tr>
<td>Saquib et al. [11]</td>
<td>64</td>
<td>Static, Dynamic [ASL, BdSL]</td>
<td>Accelerometer, Gyroscope, Flex sensor, Contact sensors</td>
<td>-</td>
<td>ANN</td>
<td>Bluetooth</td>
<td>5</td>
<td>96%</td>
</tr>
<tr>
<td>Zhang et al. [12]</td>
<td>42</td>
<td>Static, Dynamic [ASL, Traffic Command]</td>
<td>Strain sensor</td>
<td>DTW Distance</td>
<td>RBF-kernel</td>
<td>CC2530 Zigbee Module</td>
<td>3</td>
<td>94.58%</td>
</tr>
<tr>
<td>Colton et al. [27]</td>
<td>31</td>
<td>Static, Dynamic [Interaction-oriented]</td>
<td>Accelerometer, Gyroscope, Flex sensor, Pressure sensor, Magnetometer</td>
<td>Min, Max, Range, Average, Energy, Standard Deviation, Amplitude of FFT Coefficients</td>
<td>LDA, Logistic Regression</td>
<td>N/A</td>
<td>18</td>
<td>98.5%</td>
</tr>
<tr>
<td>Ariffin et al. [28]</td>
<td>14</td>
<td>Dynamic [Control Command]</td>
<td>Doppler Radar</td>
<td>-</td>
<td>DCNN</td>
<td>N/A</td>
<td>2</td>
<td>95%</td>
</tr>
<tr>
<td>Gui et al. [29]</td>
<td>26</td>
<td>Static, Dynamic [Numbers, Control Command]</td>
<td>Accelerometer, Gyroscope, Magnetometer</td>
<td>-</td>
<td>ELM</td>
<td>Bluetooth</td>
<td>2</td>
<td>89.59%</td>
</tr>
<tr>
<td>This System</td>
<td>17</td>
<td>Static, Dynamic [Interaction-oriented]</td>
<td>Accelerometer, Gyroscope, Flex sensor</td>
<td>Standard Deviation</td>
<td>KNN</td>
<td>WiFi</td>
<td>35</td>
<td>99.53%</td>
</tr>
</tbody>
</table>

**c) Use of DMP:** According to theory, orientation angles (roll and pitch) can be determined by acceleration or gyroscope. Yaw angle cannot be calculated from accelerometer data. At first, we explored the usability of raw acceleration and gyroscope data for determining the orientation angles but the gyroscope data shows drift over time due to the integration operation. On the other hand, the raw accelerometer data needs filtering. To compensate for this gyroscope drift, we tried a basic form of complementary filter fusing a small percentage of angle value from accelerometer calculation with gyroscope calculation [27], [28]. This compensated the gyroscope drift for roll and pitch angles. But yaw angles still contained a drift as we cannot calculate yaw directly from the accelerometer values. We also tried using Kalman Filter but it increased the computational complexity and run-time which was opposite to our aim [29]. Lastly, we tried the quaternions from DMP to yaw-pitch-roll conversion. This resulted in very stable signal without using any complex steps.

**d) Window size:** We have tried different window sizes for our classification task. Figure 5 shows the changes in classification accuracy with respect to window size. But there is a trade-off between window size and classification delay. The accuracy increases with the size of the window, but in real-time testing, this causes run-time delay. So, for real-time use, we have chosen the window length to be 10 samples per window which gives optimum accuracy and tolerable run-time delay.

![Window Length vs Accuracy(%)](image)

**e) Comparison with Other Similar Systems:** Several recent sensor-based hand gesture recognition works have been compared in Table III. Each system has its own merits and demerits. Moreover, as they are not based on a single standardized dataset, their performance cannot be compared merely by the recognition accuracy. Compared to other systems, our system shows greater accuracy but the number of gestures considered for this framework is less than most. On the other hand, the number of participants considered for the dataset is greater which makes it more user-independent. Moreover, we
only used one feature for detecting dynamic gestures. This results in faster detection. The work aimed to develop an optimum framework that can be used as a wireless system and have on-chip gesture detection capability. So, we tried to optimize the system rather than using more sophisticated and computationally expensive algorithms.

V. CONCLUSION

In this work, we have made a sensor-based wireless hand gesture recognition system with real-time detection capability and presented a comparative analysis of its different aspects. There are several sensor-based gesture recognition works based on custom-made data gloves and different general-purpose devices. These general-purpose devices do not fulfill the requirements in our methods. So, we had to make our robust dataglove. As the device is custom-made, there is no previous dataset based on the same hardware. So, we have built the datasets for both static and dynamic gestures. We performed necessary filtering on analog flex data. Moreover, we examined the raw accelerometer and gyroscope data closely and tried several methods to compensate gyroscope drift. Furthermore, to determine the actual motion profile, we performed gravity compensation on accelerometer data and rotate the reference frame to the NED axis. Our feature extraction and classification steps are aimed at choosing the optimum features for successful detection and real-time operation. We trained four classical machine learning classifiers. The result has shown high accuracy of detection for the gestures we chose. In future, we would like to improve our dataset by adding a sign language dictionary and include more volunteers.

REFERENCES


