SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM

JBEMi JOURNAL OF BIOMEDICAL ENGINEERING AND MEDICAL IMAGING

ISSN: 2055-1266
VOLUME 6 ISSUE 4

Development of an Infrared-Based Sensor for Finger Movement Detection

Agbotiname Lucky Imoize, Aanuoluwapo Eberechukwu Babajide
[1]*Department of Electrical and Electronics Engineering, University of Lagos, Akoka- Lagos, Nigeria*
[2]Department of Computer Engineering, University of Lagos, Akoka-Lagos, Nigeria.
aimoize@unilag.edu.ng; aanuoluwapo.babajide@live.unilag.edu. ng

ABSTRACT

With the increasing interest in smart devices and convenient remote control, the need for accurate wireless means of control has become imperative. This gives rise to the growing research interests in the area of gesture and finger movement detection. In this paper, a suitable design exploring some techniques involved in hand and finger movement detection, using depth-sensing infrared cameras embedded on Xbox Kinect Module is presented. First, 3-D images are generated and filtered along the z-axis, then two distinct techniques; Haar-Like Features, and Deep Learning using a Convolution Neural Network (CNN), are performed on the images to detect hands movement. In order to evaluate the robustness of the proposed technique, useful metrics like, Precision, Recall, F1-Score and Accuracy are used to evaluate the efficiency of the technique. Results show that while the deep learning model showed the most accurate results with a weighted accuracy of 1.0 (due to the absence of noise in the images), a weighted accuracy of 0.97 is observed for the Haar-Like features. Finally, the Haar-like features technique appears to run faster due to its static nature whereas, the deep learning model is quite slow in terms of running time. Overall, these findings point to the conclusion that the deep learning model is a better technique for detecting hands movements despite its longer running time.

Keywords**:** Finger movement detection, Haar-Like features, Deep Learning, Z-axis filtering, Accuracy, Precision, Recall, F1-score, Weighted average, 3-D Images, Depth Sensor, Infrared (IR) Images, Convolutional Neural Network (CNN).

1 Introduction

Globally, there has been an exponential growth in the number of electronic devices currently in use for various applications including fingers movement detection, and the littlest convenience could be the deciding factor in preference. Companies in the electronics industry are always open to customer-centric innovations that could give the slightest edge against competitors. This need for innovation fuels this design to create an infrared-based sensor for finger movement detection. While steering away from solutions geared towards detecting finger movement by extensive use of hardware, this design utilises the infrared depth couple in the Xbox Kinect sensor for input and then processes the generated images with a z-axis filtering algorithm, a template matching process, which uses a neural network trained feature

DOI: 10.14738/jbemi.64.7639
Publication Date: 28th June 2019
URL: http://dx.doi.org/10.14738/jbemi.64.7639

recognition model to detect and recognise captured hands. Based on the theoretical background, it is expected that successful detection and recognition will occur with good enough accuracy as compared to other methods. The response time is expected to be slower as the image processing algorithms could be computationally expensive.

In this design, infrared technology is used to detect hand and finger movements with no attached or handheld hardware piece. This solution could find several applications in gaming, vehicle control, wireless control of most electronic devices, and data analysis. More specifically, the design is aimed at detecting finger movement using infrared technology. The key objectives are; to detect hands and fingers by processing depth images from the infrared camera using Haar-like features object detection algorithm, to detect hands and fingers by processing depth images from the infrared camera using a convex hull formation and detection algorithm, to detect hands and fingers by processing depth images from the Infrared camera using a pre-trained deep learning model, and to select the best of the three algorithms using testing accuracy, precision, recall and F1 score. Furthermore, this design includes the implementation of three algorithms (Haar-like features, convex hull formation, and deep learning model) that can recognise hands and fingers given depth infrared images as input. Finally, the evaluation of the accuracy of the investigated algorithms is demonstrated to show the performances of the algorithms.

2 Related Work

Hand gestures are a common way people communicate, from simple pointing gestures to signify direction to more complex gestures in sign language. This has made a big field out of hand and finger gesture recognition as giving computers a way to recognise this basic way of communication would have many useful applications. Several researchers have made several contributions to find better ways for computers to recognise hand gestures shown by humans. Haar-like features, as proposed by Viola and Jones in [1], uses the Adaboost learning algorithm to generate classifiers by selecting critical visual features called Haar-like features in the image and then finds matches in smaller rectangular representations of the image called 'integral images'. This evolutionary method has allowed researchers to develop several object detection systems that provide good accuracy and speed.

Chen et al [2] implemented a real-time hand detection system using Haar-like features for hand posture detection and syntactic analysis based on a stochastic context-free grammar for gesture recognition. While this paper highlights the high accuracy Haar-like features offer for hand posture detection, it also points out that it cannot be used for gesture recognition because of the transition states of gestures. Hence, the use of Haar-like features in this design is limited to hand and finger detection and other static detection methods outlined in [3].

"The convex hull $C\mathcal{H}$ (S) of a set 'S' is the smallest convex set that contains 'S'[4], this statement describes a convex hull, the importance of convex hull generation is clear in the processing of images. Selecting the outer borders of an object such as a hand can detect distinct parts of the hand for further processing. Ren et al [5] used the convex hull method to detect fingers from images generated by infrared depth sensors by first using a novel method for detecting gestures called Finger Earth Mover's Distance (FEMD). After the sensor detects the hand, the generated information is converted to 'signatures' before the FEMD technique is then applied. Finger detection is achieved by either threshold decomposition or near-convex shape decomposition.

Also, Bergh et al [6] used a convex hull algorithm to get the centre of the hand in images. Infrared depth sensors were mounted on a robot that got directions in the form of pointing gestures from users. As it captures the gestures as 3D images by the sensor, their proposed model estimated the orientation of the hand by determining the wrist position regarding the centre of the hand (convex hull algorithm). After a successful orientation estimation, the model then recognised the hand posture by classifying the image based on the Average Neighbourhood Margin Minimisation (ANMM). Finally, the pointing direction was determined based on the hand posture and orientation.

Guan-Feng et al [7] used a 3D depth camera to detect the user and z-axis filtering to segment the hand. Further processing was then done on the segmented hand as a hand contour (convex hull) is formed using only the boundary pixels from the image of the hand. Gesture recognition is then done by using contour points to get convex defects in subsequent images. Evidently, the use of the convex hull algorithm allows for flexible processing because the hull provides information about the hand and different methods could draw data from the information.

Deep learning [8], as an AI technique, involves the use of a back-propagation algorithm to generate predictive models based on data supplied, thus, implying that models that detect hands can be created and trained by supplying several images of hands [9]. Oyedotun and Khashman [10] in their paper developed a system for detecting all 24 hand gestures in Thomas Moeslund's gesture recognition database [11] using a deep learning model. They also showed that more biologically oriented deep neural networks like CNN learned the hand gesture alphabet with good accuracy and relatively lower error rates.

A related work by Hussain et al [12] achieved gesture recognition by the use of VGG16 [13], a pre-trained CNN. The developed classifier recognised hand shape through transfer learning [14] over the VGG16 and traced detected hands using a skin colour detection algorithm if the motion was dynamic. Drawing a conclusion from reviewed papers, deep learning models achieve high accuracy rates for dynamic sets of data such as hand and finger movement and are hence a good technique for gesture recognition.

Discussing other techniques of hand and finger movement detection, Cheng et al [15] achieved similar results in a more minimalistic manner. Avoiding the use of a depth camera, the researchers used an Infrared LED and Infrared Receiver. This pair of sensors detected finger movement regarding time and fed the raw signal to a system that used a mathematical model in combination with a decision tree to determine the proper gesture being made. A similar but more complex approach was taken by Liu et al [16] where a Passive Infrared (PIR) sensor array was used to collect input with pseudo-random coded Fresnel lenses. The sensor arrays used compressed infrared sensing to detect only the spatial-temporal changing motion and then interpreting the information by using 'Vector Quantisation' to recognise the semantic of arm gesture.

Similarly, Megalingam et al [17] set-up an Infrared LED-Receiver sensor array embedded in a tablet and detected hand movement on the tablet by recording ulnar deviation (in the palm). With a similar use of simple infrared materials, Metzger et al [18] utilised a single Infrared proximity sensor in their wearable solution to the problem. The device, after being mounted on the ear and controlled using finger gestures, would detect an object/finger as a pulse, and multiple pulses would signify multiple fingers. The underlying technology implied that the input depended on the number of fingers that passed the sensor and not the direction of gesture as in previous methods. While these methods offer fast response time

and low computational cost and power, they, however, do not provide holistic information about the hand and fingers in this case.

Unlike the hardware solutions to gesture recognition, there are solutions that utilize truly wireless technology (mainly Infrared cameras and Infrared-based depth-sensing modules) while maintaining similar low power costs. Based on this, Erden and Çetin [19] achieved gesture recognition using a PIR sensor and an RGB camera. While the PIR sensor detected hand gesture using the Jacquard distance for the Winner Takes All (WTA) algorithm comparison, the RGB camera was to detect if the object recognised by the PIR sensor was, in fact, a hand. This approach tackled the problem of providing holistic information by separating functions and assigning them to separate sensors.

Furthermore, Ionescu et al [20, 21] used depth-sensing Infrared cameras (composed of an Infrared Designer and Infrared Receiver). The technique used was a novel space slicing technique which captured the reflected light from the sensors in 'slices' of space and calculated depth through a reconfigurable hardware unit that combined the slices by calculation to form a depth image. Further filtration of the image based on depth would then isolate the hand position in the image. Another interesting approach to the use of infrared cameras was taken by Sato et al [22] where their implementation uses a ready-made Infrared Camera as input, detects hands in a process called 'binarization', and then uses template matching (determining search windows in the image for each hand region and searching for matching fingertip templates in that region, the template used for the fingertips is a rectangle with a semi-circle at the top) for fingertip detection. The centre of the hand is found by morphological erosion operation of an extracted hand region. This method can successfully capture and detect a hand.

Xia and Fujimura [23] took a different approach in their paper where gestures are recognised using a sequence of real-time depth image data gained by the sensing hardware. The head of the user is detected using vertical and horizontal histograms while the hand of the user is detected by using depth elimination based on the aspect ratio (z-axis filtering). This approach, however, has a drawback, as it is restricted to only single users as a time. Implementations using depth cameras [24] provide high accuracy and holistic data but are computationally expensive when compared to cruder infrared methods, as they require real-time processing of feed from the depth sensor.

There also exist implementations that are uniquely different from the usual methods, one example is using an Infrared transmitter remote stick that constantly transmits a signal to a receiver placed in the vicinity [25]. The constant input from the receiver was then programmed to detect variations in the Infrared transmitter's position. The researcher then gave a set of variations that could be interpreted by the system. Hence, once a message is recognised from the input, it would initiate an action. This implementation would be faster and have good accuracy. However, it is not fully contactless as a secondary device is used for transmission. And an approach that does not make use of Infrared technology but images processing algorithms and computer vision (AI) is taken by Utsumi et al [26], where multiple RGB cameras capture the object, and the resulting images are then classified using a certain algorithm.

To make daily tasks easier and more efficient, engineering, as a mother discipline, has used a wide variety of scientific concepts. One of these applications is the case of IR technology for motion tracking. IR technology is based on the utilisation of Infrared waves. Its main application is the creation of a wide range of IR sensors. Several IR sensors exist and can range from simple IR blasters and receivers to depth-

perceptive IR cameras. Given the wide variety and their functions, how is it possible to use an IR sensor to detect the location of the hand and then the movement of fingers?

In order to solve this problem, the Infrared depth sensor couple as used by [5, 20-23, 25, 27] would be utilised. After getting the depth image, three distinct methods of image processing would obtain hand and finger movement; Haar-like features [1, 2], Convex hull detection [4-7, 28], and Deep learning model generation [9, 10, 12-14].

3 Materials and Method

The finger movement detection system presented in this study comprises of three main units; the processing unit, the input unit, the output unit. The processing unit comprises the CPU in which the code is activated, while the code provides the implementation of the designed algorithm. Finally, the processing unit interfaces with both input and output units, this makes it a central unit.

The input unit consists of the Xbox Kinect module and any other computing device used in writing the programs in the CPU. During the actual use of the system, the input unit consists of the Xbox Kinect module alone. The Xbox Kinect module is equipped with an infrared-depth couple and an RGB camera; this helps in generating the 3-D images used for finger detection. The output unit consists of a monitor screen alone, which displays the images used by the system and output of detected hands and fingers. A diagrammatic explanation of the structure is shown in Figure 1.

In this design, finger movement is achieved using an incremental model of development, as seen in Figure 2, one phase has to be completed before the next. This sub-chapter states each phase involved in the implementation phase and explains how it is done. The methods involved in implementation are software methods and would exist in codes and commands written in the Python Programming Language. Figure 3 shows the general process flow of the implementations.

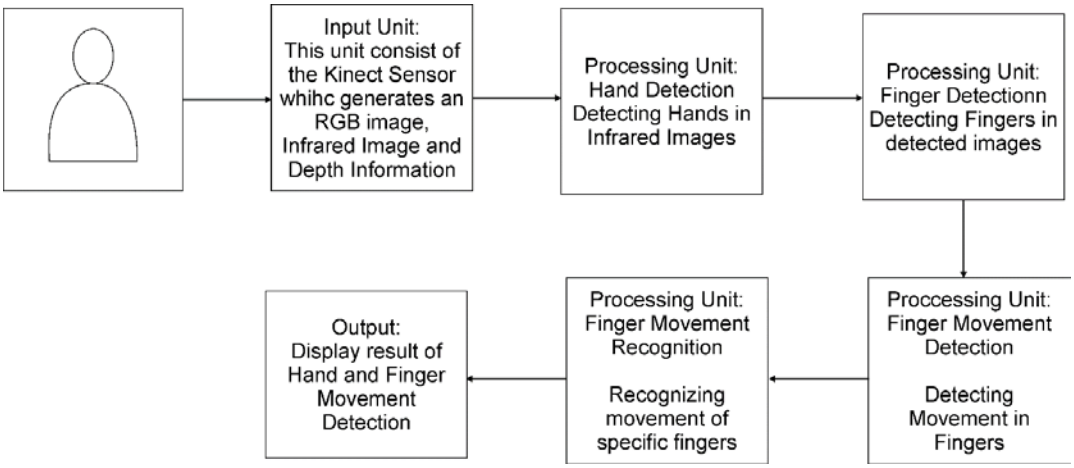

Figure 1: Physical Structural Setup of the Finger Movement Detection System

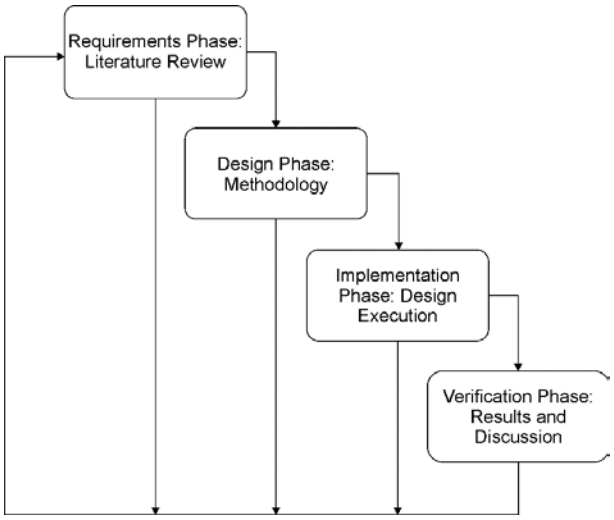

Figure 2: Incremental Development Model of Design

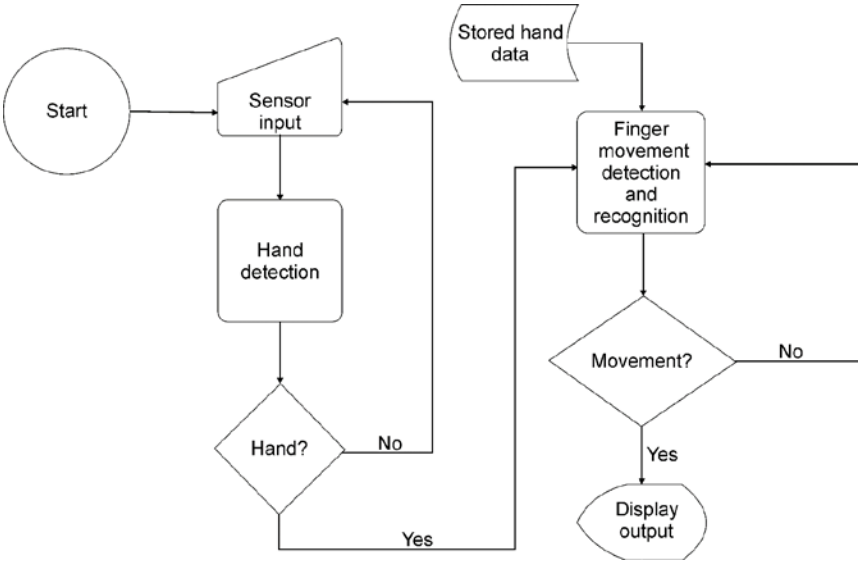

Figure 3: Process flow of Finger Movement Detection Algorithm

3.1 Setup and Configuration

To make this design work properly, it must be set up and configured properly. To configure this system, all necessary frameworks are installed; ROS [29] and Xbox Kinect 360 SDK, then communication between the sensors is established. To ensure proper synchronisation, data is transmitted from the sensor to the processing unit. The data obtained from the infrared sensor is reduced by a method called Z-Axis Filtering, which involves cutting off parts of a generated depth image beyond a specified depth. This process helps reduce the amount of information to be processed while making the hand the major object in scope. While this technique is for collecting real-time data for processing, to develop the models, a dataset of 2000 infrared images [30] was used in training and testing.

3.2 Static object detection algorithm using Haar-like Features

Considering a native RGB image, Haar-like features are generated features that are a representation of pixel intensities up to a particular location in an image, this implies that Haar-like features can be calculated fast and easily since they are only mathematical sums based on the concept of integral images [1]. Because of the speed and lightweight of Haar-like features, it is majorly used as an algorithm for image object detection, this use case is similar to this design since all hands can be classified as objects.

To use this technique, the Haar-like features algorithm is deployed to select eigenvectors from a dataset of infrared images of a hand and then store this data in a cascade (Haar-cascade). The Haar-cascade is then employed in identifying selected eigenvectors in subsequent images to determine if a hand exists or not.

3.3 Dynamic object detection using a convolutional neural network

Being born from machine learning in AI and based on ANN, deep learning is an algorithm that uses multiple layers to identify high-level features in data [31]. Deep learning algorithms can be used in any field where data of past events applies to generate predictive models with high levels of accuracy. To be implemented in this design, a deep learning model is trained to detect hands and fingers in infrared images by feeding a large enough dataset of hand images. A deep learning model is used to extract features from several sample infrared images of hands and then recognise these features in subsequent images. The model can be trained to identify hands and fingers distinctly.

3.4 Comparison of Techniques

To compare the different techniques of hand detection, it is important to select appropriate metrics that cover both strengths and weaknesses of all techniques being tested. The following detection and classification metrics have thus been selected. The confusion matrix gives a depiction of the True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN) [32]. For two sets of data, the confusion matrix is shown in equation (1). The accuracy is the ratio of correct detections to total detections; the formula is given in equation (2). The precision is the ratio of true positives to selected elements and is given in equation (3). The recall is the ratio of true positives to total positives and is given in equation (4). The F1-mean score, given in equation (5), is the harmonic mean between precision and recall. For testing under various sets of data, the weighted score, given in equation (6), gives an average score that lies heavily on the weights or count of the different sets.

$$\begin{bmatrix} TP & FN \\ TN & TN \end{bmatrix} \tag{1}$$

$$Accuracy = \frac{TN+TP}{TN+TP+FN+FP} \tag{2}$$

$$Precision = \frac{TP}{FP+TP} \tag{3}$$

$$Recall = \frac{TP}{FN+TP} \tag{4}$$

$$F1\ Mean\ Score = 2 \times \frac{precision \times recall}{precision+recall} \tag{5}$$

$$Weighted\ Average\ of\ n\ sets = \frac{\sum_{i=1}^{n} score_i \times count_i}{\sum_{i=1}^{n} count_i} \quad (6)$$

4 Results and Discussion

4.1 Static Object Detection Using Haar-Like Features

Hands were detected using Haar-Like Features, which was written in Python programming language. The results include but not limited to the successful detection of hands in different positions. Each distinct position had to be trained into the detection model as an XML file before being loaded into the python code. A sample of a real-time image is as shown in Figure 4. The python code to run the model which implements the Haar-like feature detection is wireless. Figure 5 shows the Confusion Matrix for using Haar-like features. The Precision, Recall, F1-score and weighted scores are calculated using equations (2-6) in Table 1. The confusion matrix from equation (1) generated for the results of the metrics is given in equation (7).

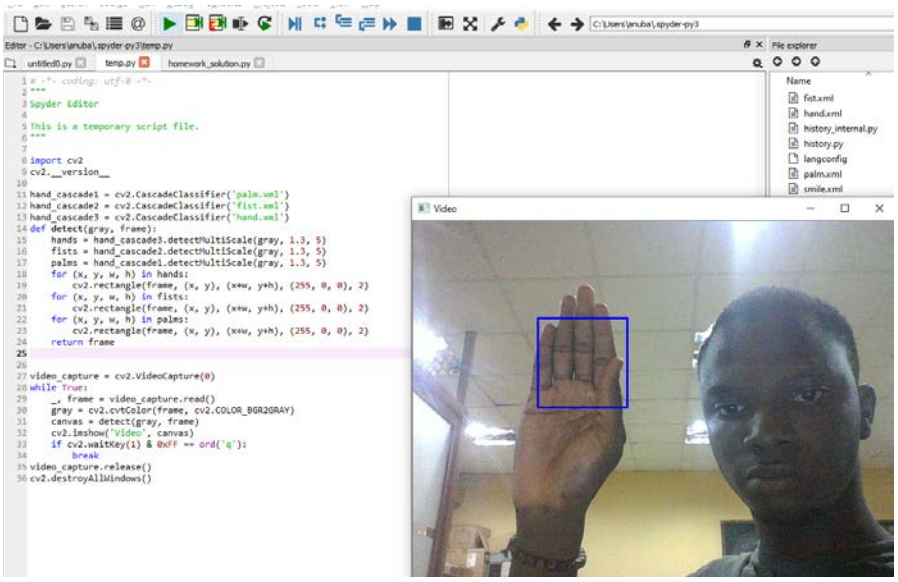

Figure 4: Hand Detection Using (Haar Cascades) Haar-Like Features

$$C_{MATRIX} = \begin{bmatrix} 208 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 210 & 0 & 0 & 0 & 6 & 0 & 1 & 0 & 0 \\ 0 & 0 & 185 & 8 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 9 & 181 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 & 182 & 3 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 0 & 188 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 180 & 0 & 4 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 196 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 5 & 0 & 213 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 \end{bmatrix} \quad (7)$$

Table 1: Metrics for Haar-Like Features

Classification Report for Haar-Like Features				
Gesture Class	Precision	Recall	F1-score	Support
Palm_Forward (1)	0.99	0.99	0.99	211
Index_Forward (2)	0.96	0.97	0.96	217
Fist (3)	0.94	0.95	0.94	195
Fist_Turned (4)	0.95	0.95	0.95	191
Thumb_Up (5)	0.99	0.97	0.98	187
Middle_Finger_Front (6)	0.95	0.97	0.96	194
OK (7)	0.97	0.97	0.97	186
Palm_Side (8)	0.98	0.98	0.98	199
C (9)	0.97	0.97	0.97	220
Point_Down (10)	1.00	1.00	1.00	200
micro avg	0.97	0.97	0.97	2000
macro avg	0.97	0.97	0.97	2000
weighted avg	0.97	0.97	0.97	2000

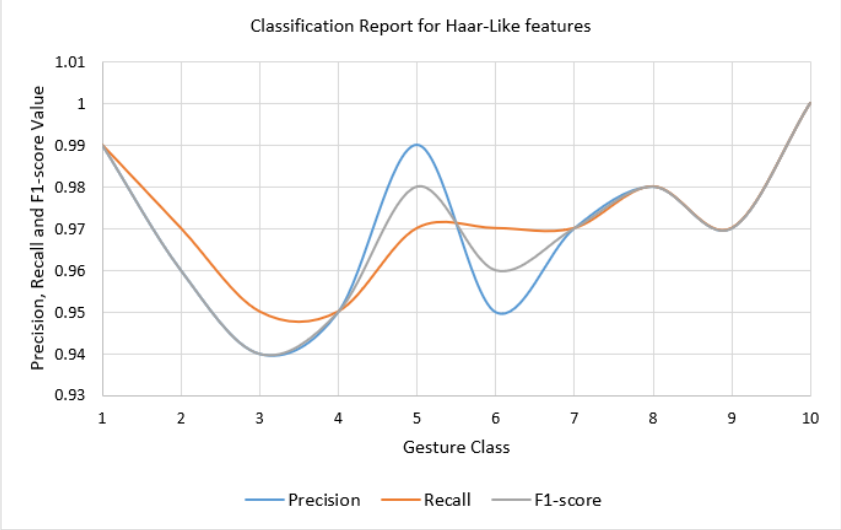

Figure 5: Scatter Plot of Precision, Recall and F1-Score for Haar-Like Features

4.2 Dynamic object detection using a convolutional neural network (CNN)

The CNN was trained using the 'leapgestrecog' dataset of 20,000 infrared images downloaded from Kaggle [30] to recognise hands under 10 different classes; Palm_Forward, Index_forward, Fist, Fist_Turned, Thumb_Up, Middle_Finger_Front, OK, Palm_side, C and Point_Down. The generated model which was trained using 10 Epochs and 14,000 steps per epoch was created using python script. Obtained results for accuracy and loss are as shown in Figure 8 and Figure 9, respectively, with their accompanying details in Table 2 and Table 3, respectively. Table 4 shows the results of the metrics (Precision, Recall, and F1-Score) of the deep learning model for each class of data. The confusion matrix generated in the form of equation (1) from the results of the metrics is as shown in equation (8). The Precision, Recall, F1-score and weighted scores are calculated as shown in Table 4, using equations (2-6).

Table 2: Training Results of CNN (Accuracy)

Model Accuracy		
Epoch	Train	Test
1	0.9311	0.9983
2	0.9975	0.9987
3	0.9989	0.9998
4	0.9982	0.9997
5	0.9995	0.9997
6	0.9994	0.9997
7	0.9989	0.9988
8	0.9996	1
9	0.9997	0.9995
10	1	1

Table 3: Training Results of CNN

Model Loss		
Epoch	Train	Test
1	0.2306	0.0066
2	0.0111	0.0061
3	0.0048	0.0003328
4	0.0107	0.0006637
5	0.002	0.003
6	0.0026	0.0037
7	0.0078	0.0082
8	0.0033	4.859E-05
9	0.0012	0.0013
10	0.0000041542	0.000091735

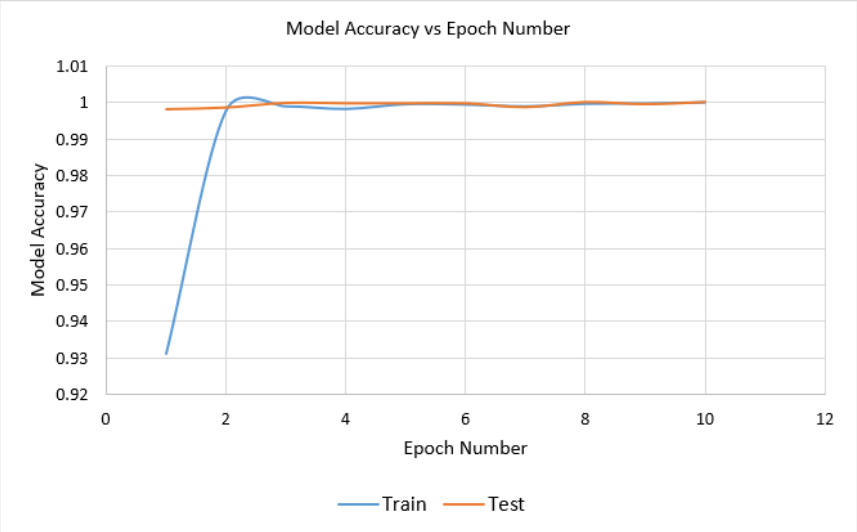

Figure 6: Graph of Model Accuracy (Accuracy vs Epoch Number)

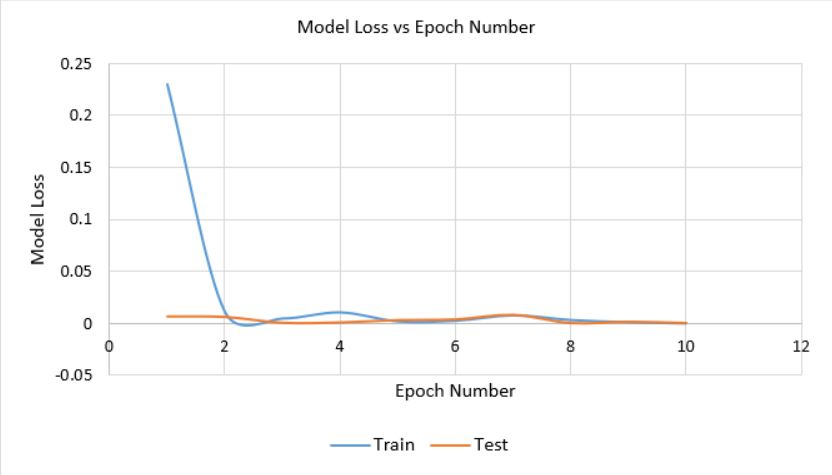

Figure 7: Graph of Model loss (loss vs Epoch Number)

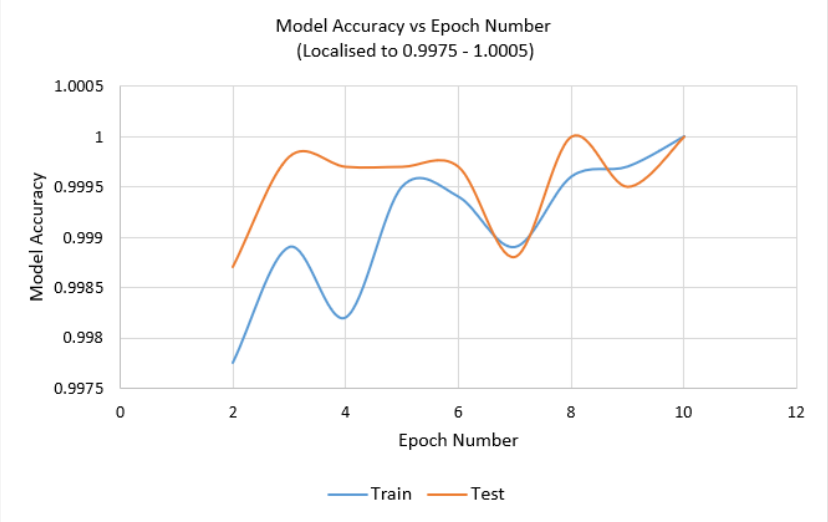

Figure 8: Graph of Model Accuracy (localised)

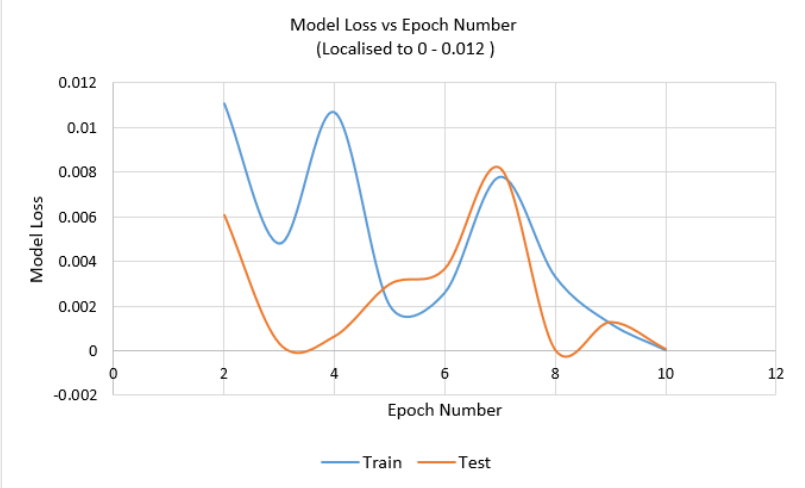

Figure 9: Graph of Model loss (localized)

$$C_{MATRIX} = \begin{bmatrix} 211 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 217 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 195 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 190 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 187 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 194 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 186 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 199 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 220 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 200 \end{bmatrix} \quad (8)$$

Table 4: Metrics for Deep Learning Model

Classification Report				
Gesture Class	Precision	Recall	F1-score	Support
Palm_Forward (1)	1.00	1.00	1.00	211
Index_Forward (2)	1.00	1.00	1.00	217
Fist (3)	0.99	1.00	1.00	195
Fist_Turned (4)	1.00	0.99	1.00	191
Thumb_Up (5)	1.00	1.00	1.00	187
Middle_Finger_Front (6)	1.00	1.00	1.00	194
OK (7)	1.00	1.00	1.00	186
Palm_Side (8)	1.00	1.00	1.00	199
C (9)	1.00	1.00	1.00	220
Point_Down (10)	1.00	1.00	1.00	200
micro avg	1.00	1.00	1.00	2000
macro avg	1.00	1.00	1.00	2000
weighted avg	1.00	1.00	1.00	2000

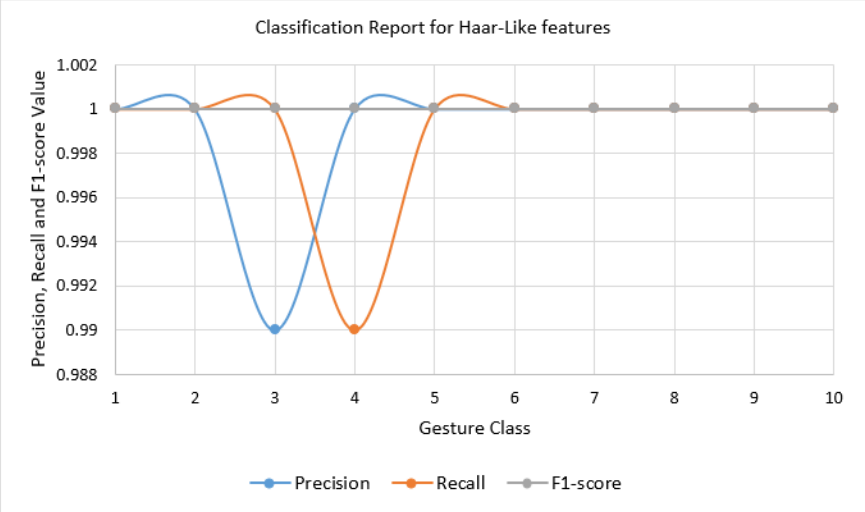

Figure 10: Scatter Plot of Precision, Recall and F1-Score for Deep Learning Model

4.3 Discussion of Results

The result of using the Haar-Cascade shows that hands are detected with small bounding boxes (Figure 4) but only when the hand is stable for a few seconds. This implies that the algorithm may not be suitable for dynamic frames or gesture recognition. Also, for each distinct position of the hand (fist, palm, palm open, two fingers up, etc.), a distinct model had to be trained to ensure detection. The confusion matrix shows the TP, TN, FN and FP's of the results when the algorithm was tested with random data. From the confusion matrix, it can be observed that the algorithm obtained the most difficulty in differentiating fists from turned fists, 'OK' gesture from 'C' gesture and 'Index_forward' from 'Middle_Finger_Front'. This can be attributed to the similarity of these gestures. Table 1 shows the calculated Precision, Recall and F1 score for the results of the test, and with an overall average of 0.97, the algorithm performs brilliantly in detecting and classifying hands, while Figure 5 gives a graphical representation of the data.

The result of using a CNN (Deep Learning Model) gave good results because the model was trained around a diverse set of infrared images and with a total of 20,000 images, the dataset was split into test and train samples. The model is trained on 12,600 samples and validated on 5,400 samples in 10 epochs, running all samples in each epoch, and the results (Accuracy and Loss) of training after each epoch can be seen in Table 2 and Table 3, respectively. From the tables, the accuracy is observed to increase for each epoch until the fourth epoch due to over fitting of the model. The model changes trend at the sixth and seventh epoch which is a good sign as it shows that the model is learning. The final three epochs show a steady increase in accuracy.

Figure 6 is a visual representation of the model accuracy at each epoch and Figure 7 is a visual representation of the model loss at each epoch, while Figure 8 and Figure 9 show the localised representations in the absence of outliers. The confusion matrix in equation (8) shows that the model is near perfect in classifying the hands under the given labels, having only one FN and FP. This is because the noise in the input data is reduced to a minimum since the images are infrared and have undergone Z-axis filtering. Table 4 gives the values of calculated Precision, Recall and F1-Score based on the Confusion Matrix, and with an overall average of 1.00, the deep learning model is perfect for predicting such data, while Figure 10 gives a graphical representation of this data.

From the results of all three techniques, the deep learning model is observed to be the best, scoring almost perfect scores in all measured metrics. While the deep learning model was slower than the static techniques, it was more accurate than both. The deep learning model has a weighted average Precision score of 1.00, weighted average recall score of 1.00 and weighted average F1-score of 1.00 while the Haar-like feature algorithm has a weighted average Precision score of 0.97, weighted average recall score of 0.97, and weighted average F1-score of 0.97.

5 Conclusion

Given the selected techniques of hand detection (Haar-like features and Deep Learning Model), and infrared images as input, the techniques are implemented in fulfilment of the set design objectives. The implementations are then evaluated based on a set of standard metrics (accuracy, precision, recall and F1-score). The results show that the deep learning model works best for hand detection and classification with a weighted average precision of 1.0, weighted average recall of 1.0, weighted average F1-score of 1.0 and accuracy of 100% mainly due to the absence of noise in the input images (infrared). Findings show

that the Haar-Like features had lower accuracy metrics than the Deep Learning model, with a weighted average precision of 0.97, weighted average recall of 0.97, weighted average F1-score of 0.97 and accuracy of 97%. However, this technique boasts a faster running time than the Deep Learning model. Overall, the Deep Learning model is comparatively favoured because the difference in speed is not noticeable enough to be a major deciding factor as opposed to the difference in terms of accuracy.

Future work would consider the development of a suitable hybrid model that has the accuracy of the deep learning model and the speed of the Haar-like features model for optimal results at all times. In addition, due to limited interest in this field, available datasets with holistic information (infrared and colour depth images alongside accurate detection labels and classes) are grossly insufficient for larger scaled modelling. Hence, there is a need for accurate and holistic datasets of infrared depth images of hands specifically for training and testing models that would achieve better results.

REFERENCES

[1] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision,* vol. 4, pp. 137-154, 2001.

[2] Q. Chen, N. D. Georganas, and E. M. Petriu, "Hand Gesture Recognition Using Haar-Like Features and a Stochastic Context-Free Grammar," *IEEE Transactions on Instrumentation and Measurement,* vol. 57, pp. 1562-1571, 2008.

[3] S. Sharma and S. Jain, "A Static Hand Gesture and Face Recognition System for Blind People," in *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)*, 2019, pp. 534-539.

[4] M. De Berg, M. Van Kreveld, M. Overmars, and O. Schwarzkopf, "Computational geometry," in *Computational geometry*, ed: Springer, 1997, pp. 1-17.

[5] Z. Ren, J. Meng, and J. Yuan, "Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction," in *2011 8th International Conference on Information, Communications & Signal Processing*, Singapore., 2011., pp. 1-5.

[6] M. V. d. Bergh, D. Carton, R. D. Nijs, N. Mitsou, C. Landsiedel, K. Kuehnlenz*, et al.*, "Real-time 3D hand gesture interaction with a robot for understanding directions from humans," in *2011 RO-MAN*, Atlanta, GA., 2011., pp. 357-362.

[7] H. Guan-Feng, K. Sun-Kyung, S. Won-Chang, and J. Sung-Tae, "Real-time gesture recognition using 3D depth camera," in *2011 IEEE 2nd International Conference on Software Engineering and Service Science*, Beijing, China., 2011., pp. 187-190.

[8] C. F. Higham and D. J. Higham, "Deep Learning: An Introduction for Applied Mathematicians," *SIAM Review,* vol. 61, pp. 860-891, 2019.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature,* vol. 521, pp. 436-444, 2015.

[10] O. K. Oyedotun and A. Khashman, "Deep learning in vision-based static hand gesture recognition," *Neural Computing and Applications,* vol. 28, pp. 3941-3951, December 01 2017.

[11] H. Birk, T. B. Moeslund, and C. B. Madsen, "Real-time recognition of hand alphabet gestures using principal component analysis," in *Proceedings of the Scandinavian conference on image analysis*, 1997, pp. 261-268.

[12] S. Hussain, R. Saxena, X. Han, J. A. Khan, and H. Shin, "Hand gesture recognition using deep learning," in *2017 International SoC Design Conference (ISOCC)*, Seoul, South Korea., 2017., pp. 48-49.

[13] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[14] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering,* vol. 22, pp. 1345-1359, 2009.

[15] H. Cheng, A. M. Chen, A. Razdan, and E. Buller, "Contactless gesture recognition system using proximity sensors," in *2011 IEEE International Conference on Consumer Electronics (ICCE)*, Berlin, Germany., 2011., pp. 149-150.

[16] T. Liu, X. Luo, J. Liu, and H. Cui, "Compressive infrared sensing for arm gesture acquisition and recognition," in *2015 IEEE International Conference on Information and Automation*, Lijiang, China., 2015., pp. 1882-1886.

[17] R. K. Megalingam, V. Rangan, S. Krishnan, and A. B. E. Alinkeezhil, "IR Sensor-Based Gesture Control Wheelchair for Stroke and SCI Patients," *IEEE Sensors Journal,* vol. 16, pp. 6755-6765, 2016.

[18] C. Metzger, M. Anderson, and T. Starner, "FreeDigiter: a contact-free device for gesture control," in *Eighth International Symposium on Wearable Computers*, Arlington., 2004., pp. 18-21.

[19] F. Erden and A. E. Çetin, "Hand gesture based remote control system using infrared sensors and a camera," *IEEE Transactions on Consumer Electronics,* vol. 60, pp. 675-680, 2014.

[20] D. Ionescu, V. Suse, C. Gadea, B. Solomon, B. Ionescu, and S. Islam, "An infrared-based depth camera for gesture-based control of virtual environments," in *2013 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, Milan, Italy., 2013., pp. 13-18.

[21] D. Ionescu, V. Suse, C. Gadea, B. Solomon, B. Ionescu, S. Islam*, et al.*, "A single sensor NIR depth camera for gesture control," in *2014 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*, Montevideo, Uruguay., 2014., pp. 1600-1605.

[22] Y. Sato, Y. Kobayashi, and H. Koike, "Fast tracking of hands and fingertips in infrared images for augmented desk interface," in *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, Grenoble, France., 2000, pp. 462-467.

[23] L. Xia and K. Fujimura, "Hand gesture recognition using depth data," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, 2004, pp. 529-534.

[24] J. Suarez and R. R. Murphy, "Hand gesture recognition with depth images: A review," in *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, Paris, France., 2012, pp. 411-417.

[25] H. Ruser, S. Kosterski, and C. Kargel, "Gesture-based universal optical remote control: Concept, reconstruction principle and recognition results," in *2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*, Pisa, Italy., 2015, pp. 1677-1681.

[26] A. Utsumi, N. Tetsutani, and S. Igi, "Hand detection and tracking using pixel value distribution model for multiple-camera-based gesture interactions," in *Proceedings. IEEE Workshop on Knowledge Media Networking*, Anacapri, Italy., 2002, pp. 31-36.

[27] Z. Zhang, "Microsoft Kinect Sensor and Its Effect," *IEEE MultiMedia,* vol. 19, pp. 4-10, 2012.

[28] Y. Li, "Hand gesture recognition using Kinect," in *2012 IEEE International Conference on Computer Science and Automation Engineering*, 2012, pp. 196-199.

[29] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs*, et al.*, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, Japan., 2009, p. 5.

[30] Kaggle, "Hand Gesture Recognition Database," GTI, Ed., ed. kaggle.com: kaggle.com, 2018.

[31] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks,* vol. 61, pp. 85-117, 2015.

[32] A. L. Imoize, T. Oyedare, M. E. Otuokere, and S. Shetty, “Software Intrusion Detection Evaluation System: A Cost-Based Evaluation of Intrusion Detection Capability,” Communications and Network, vol. 10, no. 4, pp. 211–229, Oct. 2018.

