3.3.6 SKIN DETECTION:

The skin colour detection is one of important goal in hand gesture recognition. Skin colour detection decision rules which we have to build that will discriminate between skin portion and non-skin portion pixels. This is accomplished usually by metric introduction, which measure distance of the pixel colour. This metric type is knows as skin modeling.

3.3.7 EXPLICITLY DEFINED SKIN REGION

Following are some common ethnic skin groups and there RGB color space:

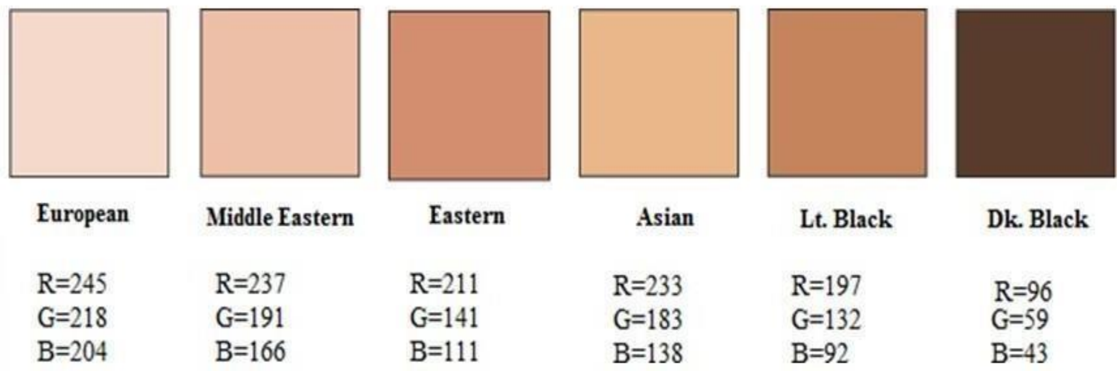


Fig 3.2: Different Ethnic Group Skin Patches

To build a skin classifier is to define explicitly through a number of rules the boundaries of skin color cluster in some color space. The advantage of this method is the simplicity of skin detection rules that leads to the construction of very rapid classifier. For Example (R,G,B) is classified as skin if:

$R > 95$ and $G > 40$ and $B > 20$ and $\max\{R,G,B\}-\min\{R,G,B\} > 15$ and

$|R-G| > 15$ and $R > G$ and $R > B$

In this classifier threshold defined to maximize the chance for recognizing the skin region for each colour. If we see in Figure 3.2 that Red colour in every skin sample is greater than 95, Green is greater than 40 and Blue is greater than 20 in**.** So threshold can make this classifier easily detect almost all kind of skin.

This is one of the easiest methods as it explicitly defines skin-color boundaries in different color spaces. Different ranges of thresholds are defined according to each color space components in as the image pixels that fall between the predefined ranges are considered as skin pixels. The advantage of this method is obviously the simplicity which normally avoids of attempting too complex rules to prevent over fitting data. However, it is important to select good colour space and suitable decision rules to achieve high recognition rate with this method.

3.3.8 REMOVAL OF BACKGROUND

I have found that background greatly affects the results of hand detection that's why I have decided to remove it. For this I have written our own code in spite of using any built-in ones.

Before **After**

Fig: 3.3: Removal of Background

3.4 CONVERSION FROM RGB TO BINARY

All algorithms accept an input in RGB form and then convert it into binary format in order to provide ease in recognizing any gesture and also retaining the luminance factor in an image.

3.4.1 HAND DETECTION

Image could have more than one skin area but we required only hand for further process. For this I choose criteria image labeling which is following

3.4.2 LABELING

To define how many skin regions that we have in image is by labeling all skin regions. Label is basically an integer value have 8 connecting objects in order to label all skin area pixel. If object had label then mark current pixel with label if not then use new label with new integer value. After counting all labeled region (segmented image) I sort all them into ascending order with maximum value and choose the area have maximum value which I interested because I assume that hand region in bigger part of image. To separate that region which looked for, create new image that have one in positions where the label occurs and others set to zero.

Before **After**

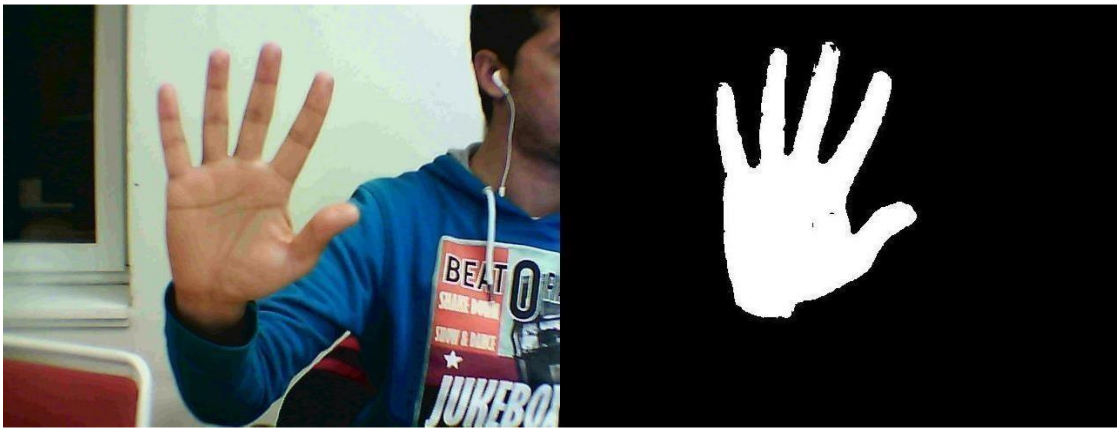


Fig 3.4: Labeling Skin Region

3.5 FEATURE EXTRACTION ALGORITHMS

There are four types of algorithms that I studied and implemented namely as followings:

- Row vector algorithm
- Edging and row vector passing
- Mean and standard deviation of edged image
- Diagonal sum algorithm

3.6 REAL TIME CLASSIFICATION

Fig 3.5 shows the concept for real time classification system. A hand gesture image will be passed to the computer after being captured through camera at run time and the computer will tyro recognize and classify the gesture through computer vision.

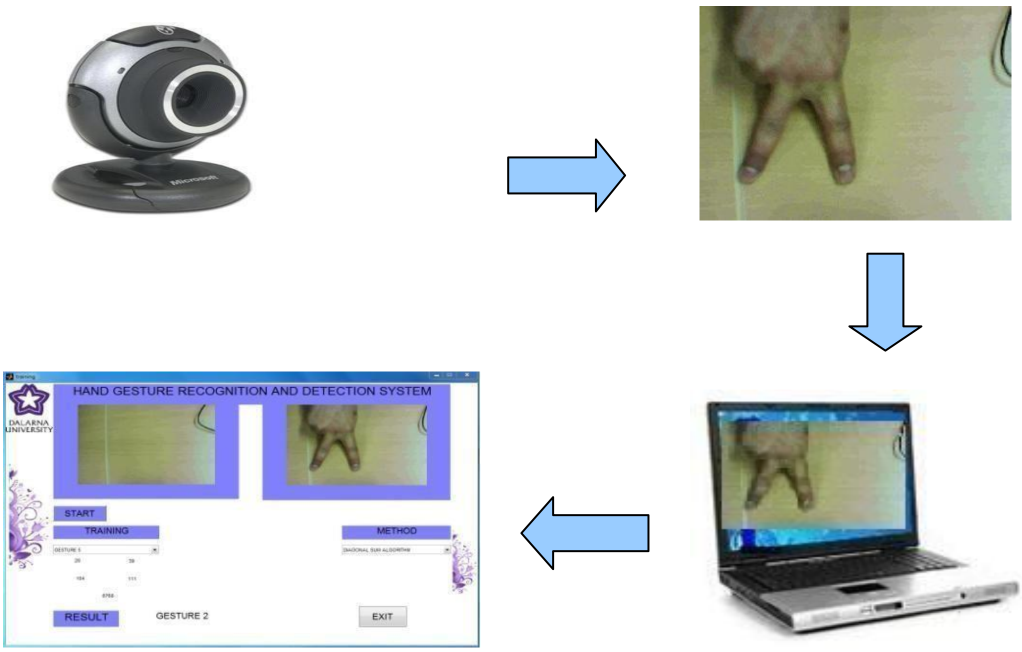


Fig 3.5: Real Time Classification

In real time classification the system developed tries to classify the gestures not saved before but given at the run time. The system first trains itself with the user count gestures at the run time and then tries to classify the newly given test gestures by the user. The algorithm used by the system for real time classification is the diagonal sum algorithm.

CHAPTER-4

PRELIMINARY ANALYSIS

4.1 NEURAL NETWORKS

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. We can train a neural network to perform a particular function by adjusting the values of the connections (weights) between elements.

Commonly neural networks are adjusted, or trained, so that a particular input leads to a specific target output. Such a situation is shown in Fig 4.1 below. There, the network is adjusted, based on a comparison of the output and the target, until the network output matches the target.

Typically many such input/target pairs are used, in this supervised learning to train a network .

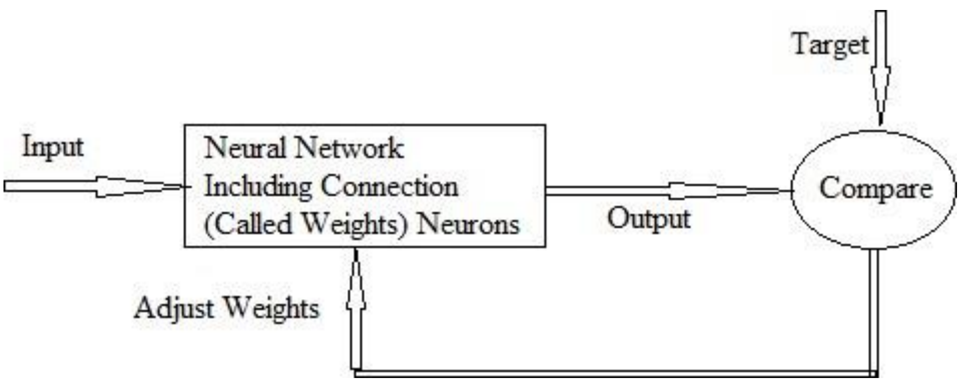


Fig 4.1 Neural net Block Diagram

Neural networks have been trained to perform complex functions in various fields of application including pattern recognition, identification, classification, speech, and vision and control systems. Today neural networks can be trained to solve problems that are difficult for conventional computers or human beings.

Once data ready for representation then next step is to design NN for training and testing data. In first two algorithms Row Vector and Edging and Row Vector passing algorithm have three layers feed forward network: Input, Hidden and Output. Number of neuron in Input is 640 which are equal to number of features extracted from each of algorithm and one neuron for Output layer for skin class to be recognized. But for Mean and standard deviation there are only two input which is also equal to extracted features from this algorithm. Neural network Architecture has number of parameter such as learning rate (lr), number of epochs and stopping criteria which is based on validation of data. Training of Mean Square Error at output layer which is set trial values and which is set by several experiments.

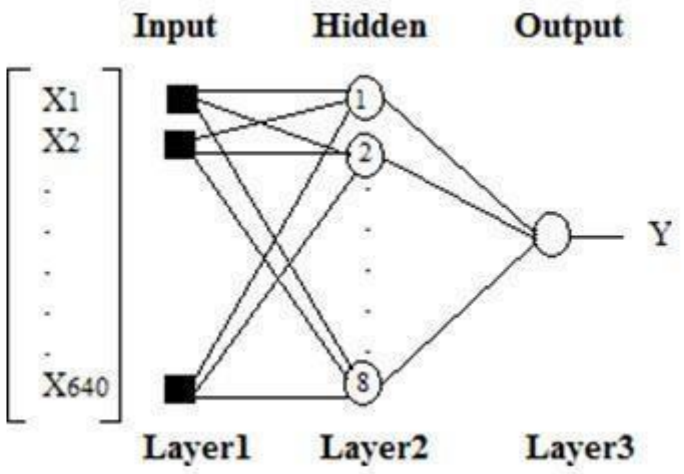


Fig 4.2: NN for Row Vector and Edging Row Vector

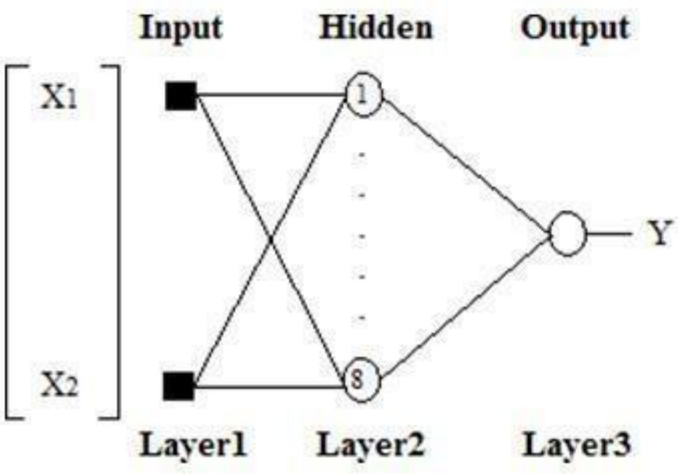


Fig 4.3: NN for Mean and S.D

4.2 ROW VECTOR ALGORITHM

We know that behind every image is a matrix of numbers with which we do manipulations to derive some conclusion in computer vision. For example we can calculate a row vector of the matrix. A row vector is basically a single row of numbers with resolution 1*Y, where Y is the total no of columns in the image matrix. Each element in the row vector represents the sum of its respective column entries as illustrated in Fig 4.4:

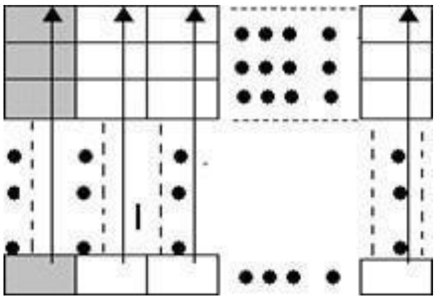


Fig 4.4 Row vector of an image

The first algorithm I studied and implemented makes use of the row vector of the hand gestures. For each type of hand gesture, I took several hand images, do skin modeling, labeling, removed their background and RGB to binary conversion in the preprocessing phase, calculated their row vectors and then trained the neural network with these row vectors. Ultimately, the neural network was able to recognize the row vectors that each gesture count can possibly have. Hence, after training, the system was tested to see the recognition power it had achieved.

Mathematically, we can describe the image for training or testing purpose given to the neural network as:

Input to neural network = Row vector (After image Preprocessing)

The flowchart of the algorithm is given below in Fig 4.5:

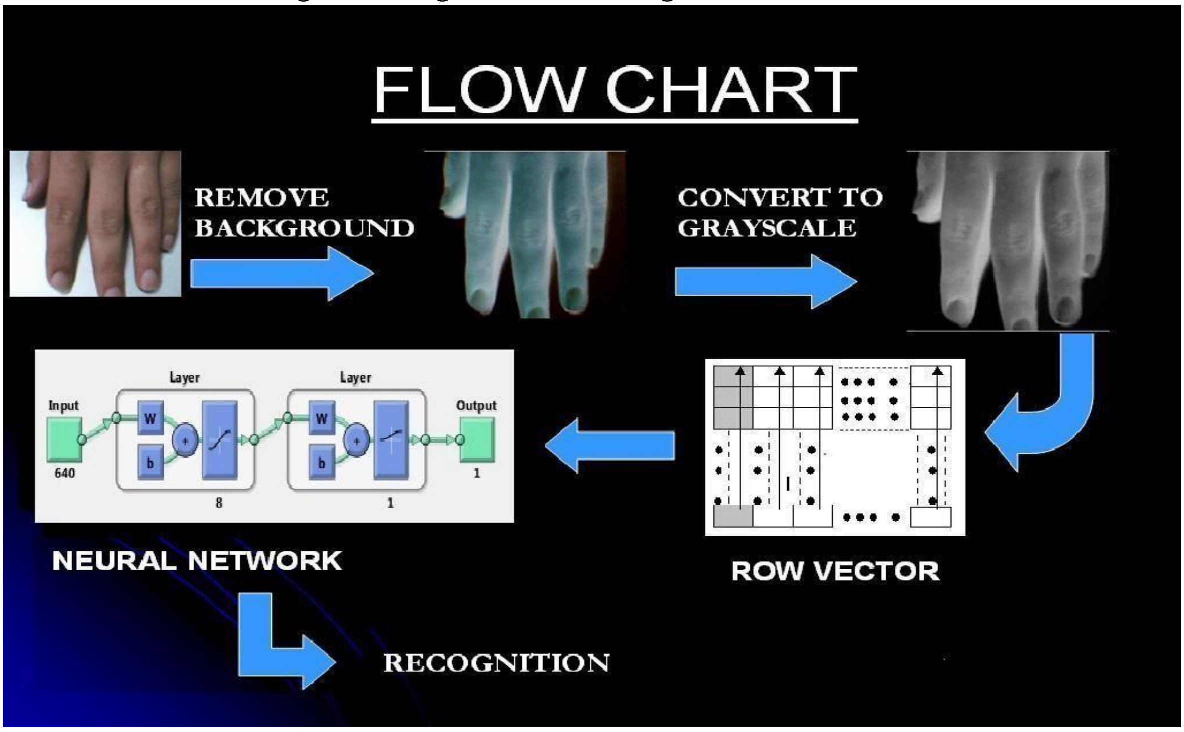


Fig 4.5: Row Vector Flow Chart

4.3 EDGING AND ROW VECTOR PASSING ALGORITHM

In the pre-processing phase of this algorithm, I do preprocess, skin modeling and removed the background etc. of the gesture image taken. This image was converted from RGB into grayscale type. Gray scale images represent an image as a matrix where every element has a value corresponding to how bright/dark the pixel at the corresponding position should be colored. For representing the brightness of pixels there are two ways for represent numbers, First class called Double class that assign floating numbers ("decimals") between 0 and 1 for each pixel. The zero (0) value represent black and value one (1) corresponds to white. The second class known as unit8 that assign integer between 0 and 255 for brightness of pixel, zero (0) correspond to black and 255 for white. The unit8 class requires less storage than double roughly 1/8.

After the conversion of the image into grayscale, I took the edge of the image with a fixed threshold i.e. 0.5.This threshold helped us in removing the noise in the image. In the next step, a row vector of the edged image was calculated. This row vector is then passed on to the neural network for training. The neural network (NN) is later on tested for the classification of the gestures.

Mathematically, the input to the neural network is given as:

Input to NN= Row vector [Edge (Grayscale image)]

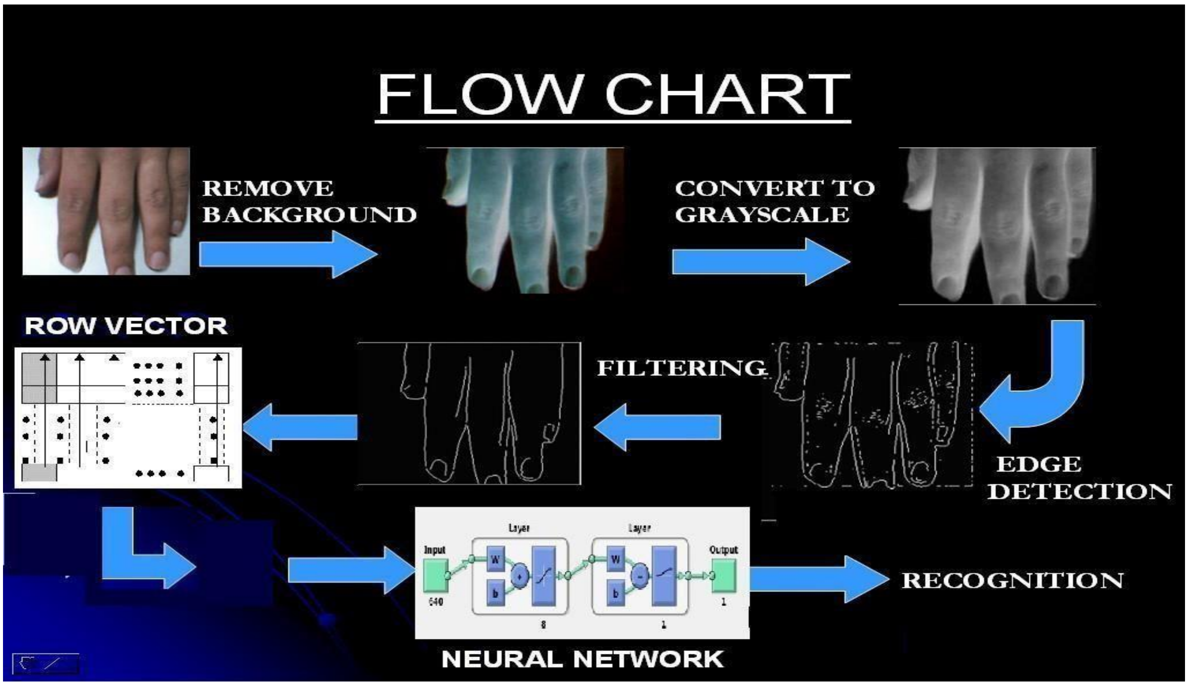


Fig 4.6: Edging and Row Vector Flow Chart

4.4 MEAN AND STANDARD DEVIATION OF EDGED IMAGE

In the pre-processing phase, doing several step like removing the background and RGB image is converted into grayscale type as done in the previous algorithm. The edge of the grayscale image is taken with a fixed threshold i.e. 0.5 then calculate the mean and standard deviation the processed image.

Mean is calculated by taking a sum of all the pixel values and dividing it by the total no of values in the matrix. Mathematically, it is defined as:

$$\overline{X} = \sum_{i=1}^{n} Xi / n \qquad (4.1)$$

Stand Deviation can calculate from mean which is mathematically defined as:

$$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \overline{x})^2}{n}} \qquad (4.2)$$

The mean and standard deviation of each type of count gesture are given to the neural network for training. In the end, the system is tested to see the success rate of classification this technique provides. Mathematically, the input given to the neural network is defined as:

Input to NN= Mean (Edge (Binary image)) + S.D (Edge (Binary Image))

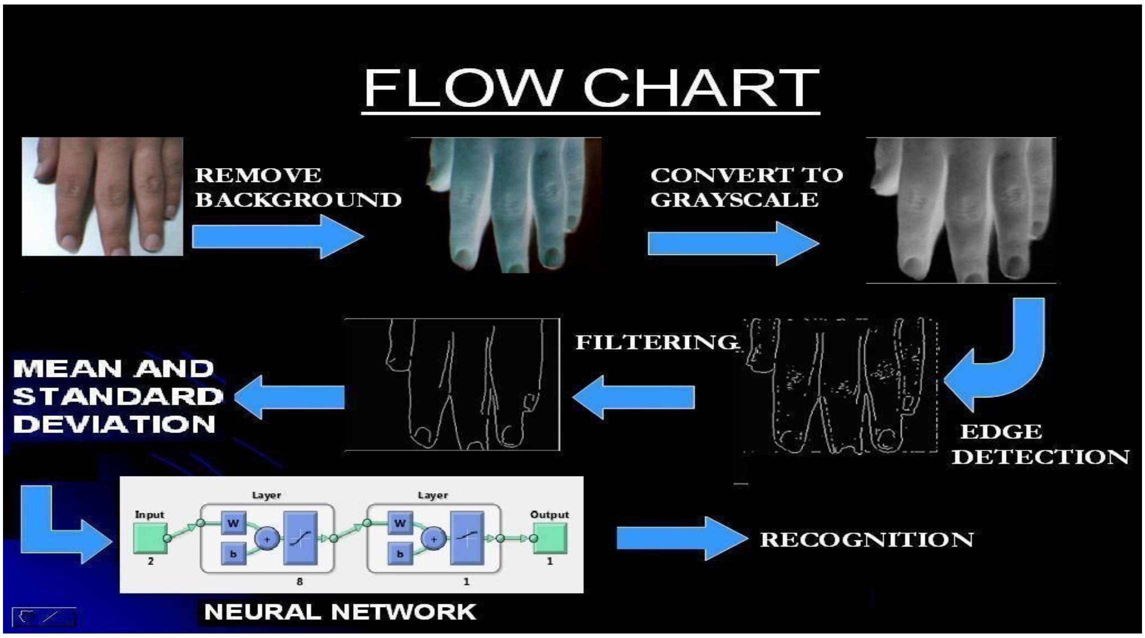


Fig 4.7: Mean & S.D Flow Chart

4.5 DIAGONAL SUM ALGORITHM

In the pre-processing phase, doing mentioned steps in methodology, skin modeling removal of the background, conversion of RGB to binary and labeling. The binary image format also stores an image as a matrix but can only colour a pixel black or white (and nothing in between).

It assigns a 0for black and a 1 for white. In the next step, the sum of all the elements in every diagonal is calculated. The main diagonal is represented as k=0 in Figure 4.7 given below; the diagonals below the main diagonal are represented by k<0 and those above it are represented as k>0

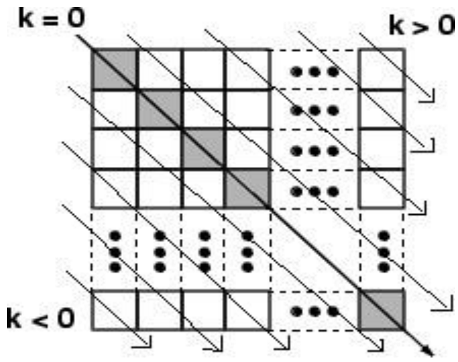


Fig: 4.8: Diagonal Sum

The gesture recognition system developed through this algorithm first train itself with the diagonals sum of each type of gesture count at least once, and then its power could be tested by providing it with a test gesture at real time. Mathematically, the input given to the system at real time is given as:

$$X_i = \sum_{i=1}^{n} Diagonals$$
$$\text{Input} = \sum_{i=1}^{n} X_i \quad (4.3)$$

The flowchart of the algorithm is given below in Fig 4.9:

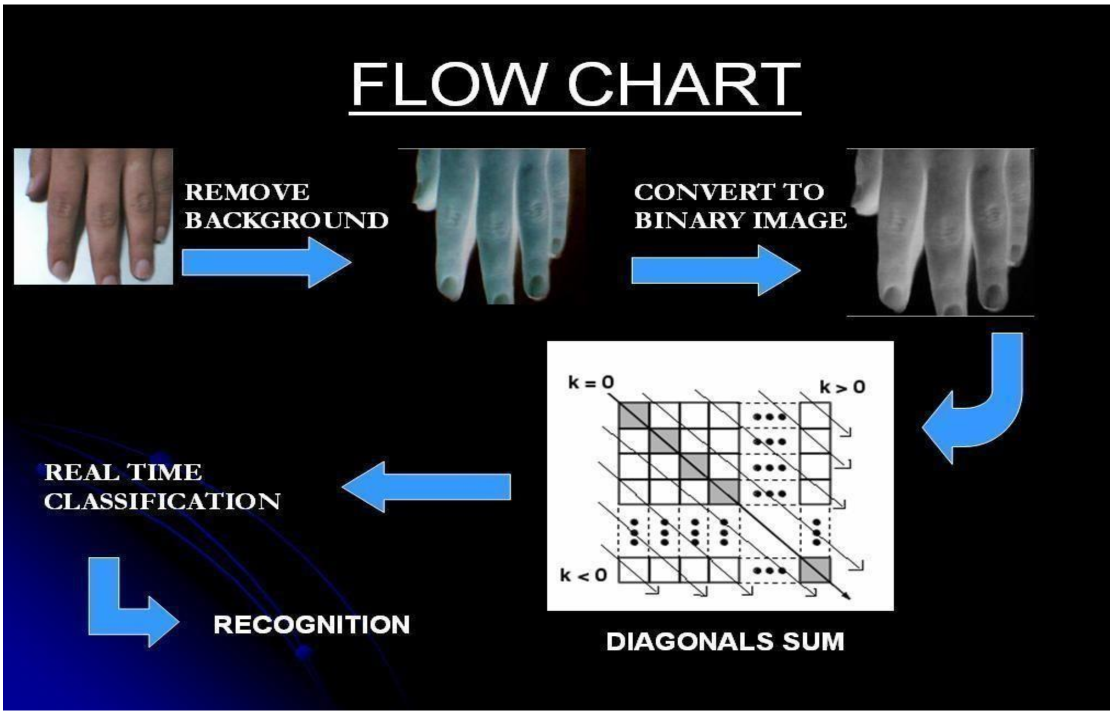


Fig 4.9: Diagonal Sum Flow Chart

4.6 GRAPHICAL USER INTERFACE (GUI)

GUIDE is MATLAB's Graphical User Interface Development Environment. GUIDE use for GUIs containing various style figure windows of user interface objects. For creating GUI each object must be programmed to activate user interface.

4.7 GUI DESIGN

The next stage was to design a GUI such that it reflected the GUI requirements stated above. Following Fig 4.9 shows the GUI design:

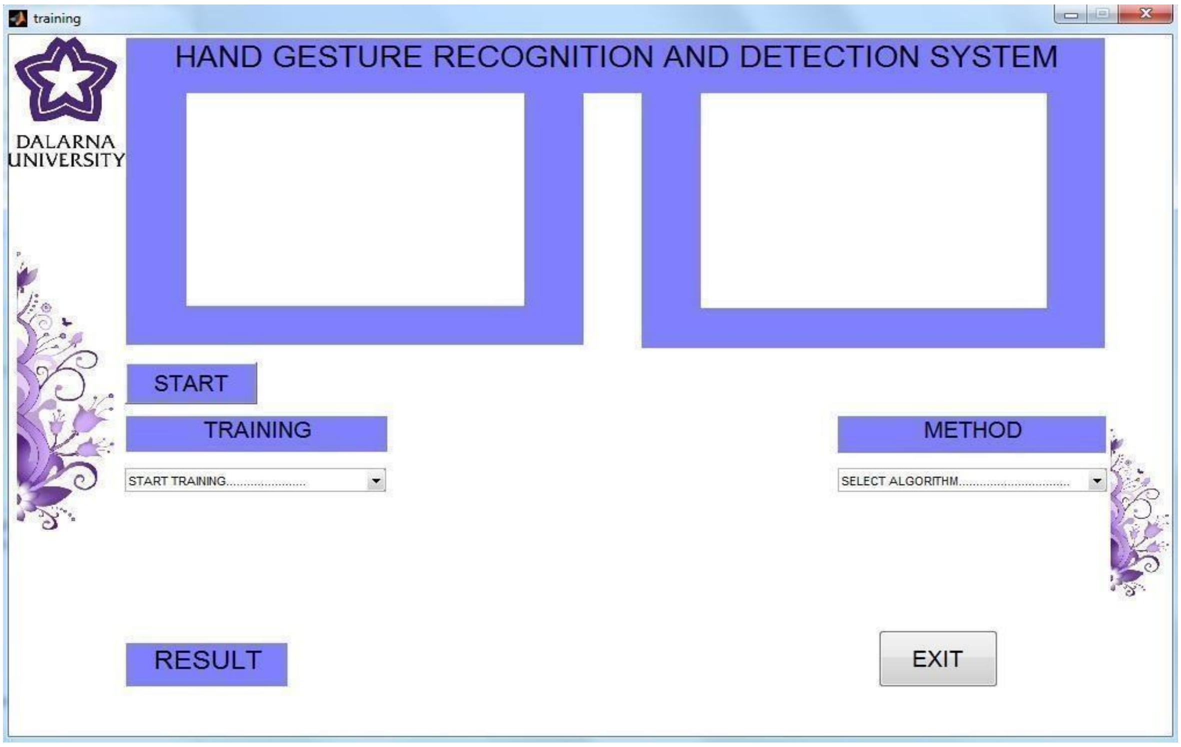


Fig 4.10: Graphical User Interface

START button activates the web cam. User can view his/her hand in the first box, which is above the start button. On selecting any gesture from drop down list under the TRAINING button, image will be captured and displayed in the right-hand box.

My first step is training for which we capture different images and select respective options of gesture numbers from the drop down menu in order to train the system. When an option is chosen from the drop down menu, the user is asked to enter a name for the training image. Recognition process can now be started by capturing a test gesture and then clicking the any of algorithm under METHOD button. This displays a save window that stores your test gesture by the name you give it. After that a progress bar indicates the processing of the system (i.e. pre-processing and recognition phase). The result of the system will appear in front of the RESULT textbox. EXIT button enables the user to quit the MAT LAB.

4.7.1 NEURAL NETWORK TRAINING IF NN ALGORITHM SELECTED:

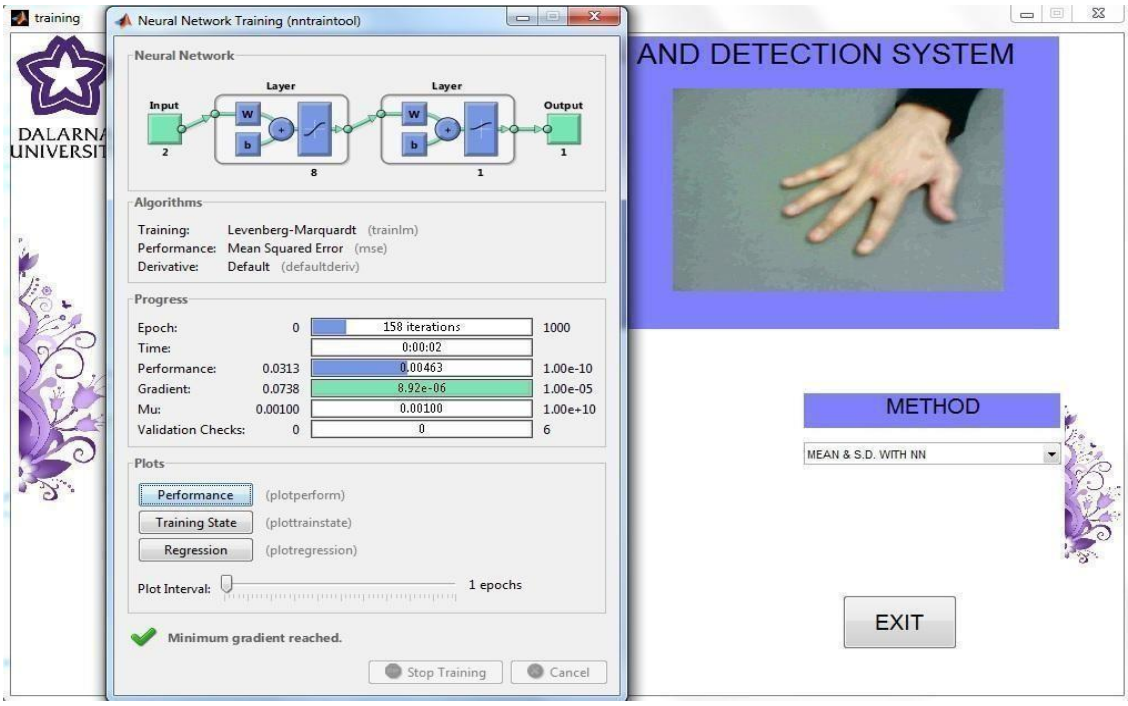


Fig 4.11: NN Training

This neural network training system will pop up when we select Row Vector algorithm, Edging and Row Vector and Mean and Standard Deviation algorithm. This is not for Diagonal Sum algorithm. Diagonal Sum classified real time.

4.7.2 PERFORMANCE OF NN

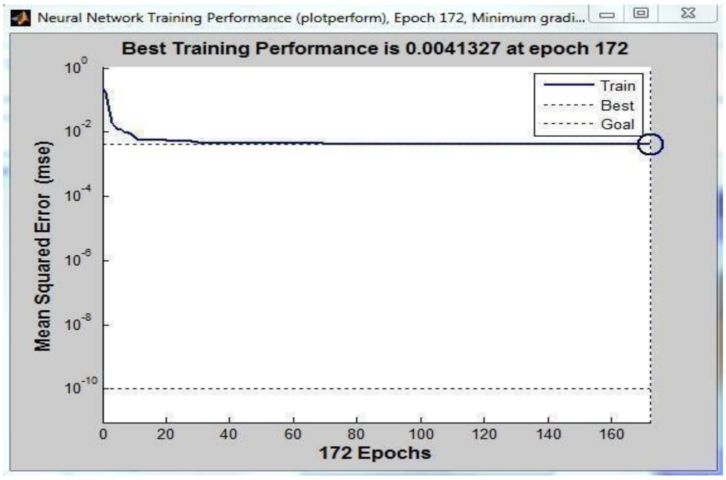


Fig: 4.12: Performance Chart

4.7.3 DETECTION AND RECOGNITION OF A GESTURE

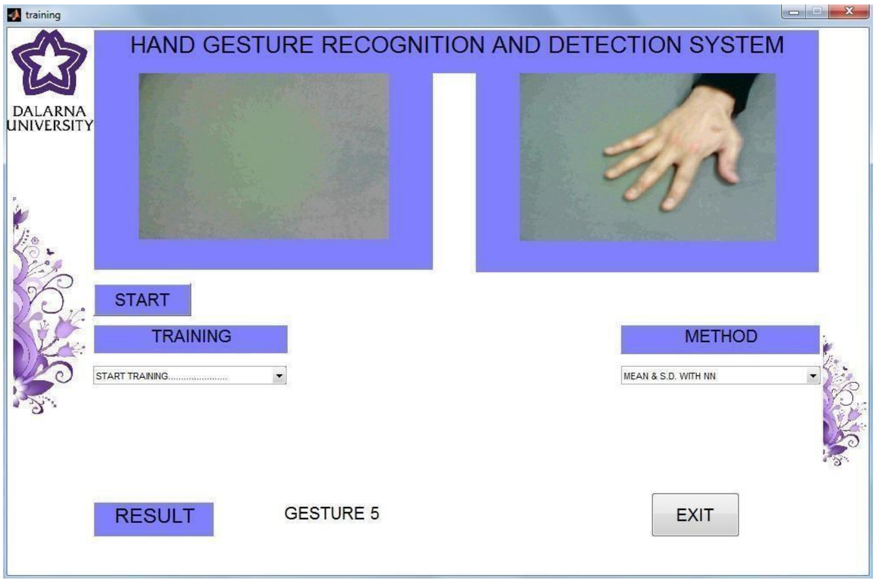


Fig: 4.13: Graphical User Interface Output

For Diagonal Sum algorithm we need to select 5 different gestures by selecting drop down menu under TRAINING for real time training and for recognition select Diagonal Sum Algorithm under METHOD drop down menu. There is no neural network for this algorithm but remaining 3 algorithms have neural network for train the system.

4.8 RESULTS AND DISCUSSION

The hand gesture recognition system has been tested with hand images under various conditions. The performance of the overall system with different algorithms is detailed in this chapter. Examples of accurate detection and cases that highlight limitations to the system are both presented, allowing an insight into the strengths and weaknesses of the designed system. Such insight into the limitations of the system is an indication of the direction and focus for future work. System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. It helps us in uncovering errors that were made inadvertently as the system was designed and constructed. We began testing in the „small" and progressed to the „large". This means that early testing focused on algorithms with very small gesture set and we ultimately moved to a larger one with improved classification accuracy and larger gesture set.

4.9 ROW VECTOR ALGORITHM

The detection rate of the system achieved through this algorithm was 39%. It was noticed that the performance of the system improved as the data set given to neural network (NN) for training was increased. For each type of gesture, 75 images were given to the system for training. At the end, the system was tested with 20 images of each kind of gesture. The results of the algorithm are given below in Fig 4.14.

The row vector algorithm failed to give satisfactory results because the parameter (row vector) of two different pictures happened to be the same for different gestures. This resulted in wrong classification of some of the gestures and also it takes too much time in training process so, a need was felt for improvement in the parameter passed to the neural network (NN). This resulted in the evolution of my edge and row vector-passing algorithm.

4.10 EDGING AND ROW VECTOR PASSING ALGORITHM

The detection rate of the system achieved through this algorithm was 47%. It was noticed that the performance of the system improved as the data set for training was increased. For each type of gesture, 75 images were given to the system for training. At the end, the system was tested with 20 images of each kind of gesture. The results of the algorithm are given below in Fig 4.14.

The introduction of edge parameter along with the row vector gained an improvement in performance but the self-shadowing effect in edges deteriorated the detection accuracy and it was again thought to improve the parameter quality passed to the neural network (NN). It also have drawback of time consuming it take more than normal time for training process. This gave birth to mean and standard deviation of edged image algorithm.

4.11 MEAN AND STANDARD DEVIATION OF EDGED IMAGE

The detection rate of the system achieved through this algorithm was 67%. It was noticed that the performance of the system improved as the data set for training was increased. For each type of gesture, 75 images were given to the system for training. At the end, the system was tested with 20 images of each kind of gesture. The implementation details of the algorithm are given below in Fig 4.14. The mean and standard deviation algorithm did help us in attaining an average result and also it take less time for training process but still the performance was not very good as I want. The reason was majorly the variation in skin colours and light intensity.

4.12 DIAGONAL SUM ALGORITHM

The poor detection rate of the above algorithms resulted in the evolution of diagonal sum algorithm, which used the sum of the diagonals to train and test the system. This is real time classification algorithm. User need to train system first and then try to recognized gesture. Every gesture at least once user should have to give system for training process. The detection rate of the system achieved through this algorithm was 86%. For each type of gesture, multiple images were given to the system for training. After every training process system were tested 20 times. At the end, the system was tested with 20 images of each kind of gesture. The results of the algorithm are given below in Fig 4.14. The diagonal sum algorithm also demanded improvement as its detection accuracy was not 100% but it was good.

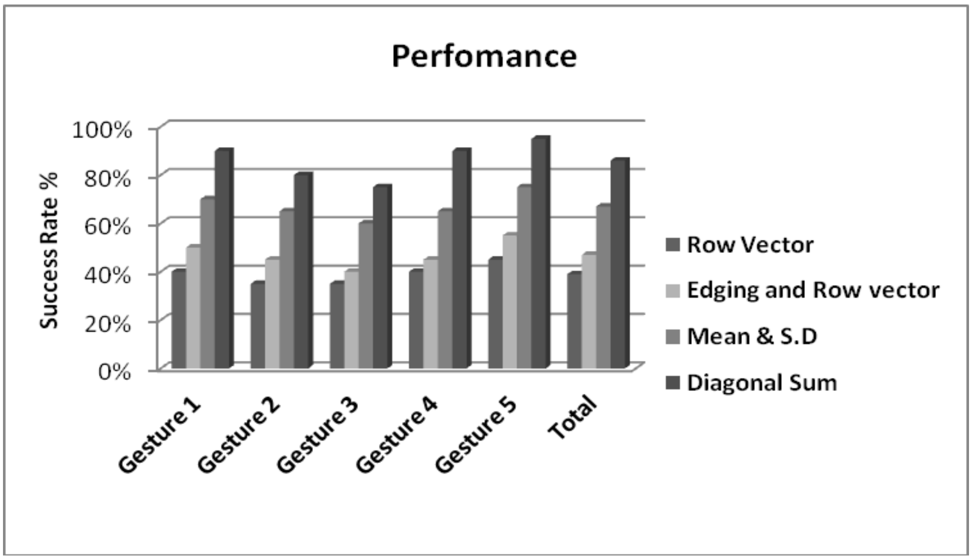


Fig 4.14 : Performance Percentage

4.13 PROCESSING TIME

Evaluation of time in an image processing and recognition procedure is very important for result and performance, which shows tendency in all techniques which I used to recognized hand gestures. There are few factors that prejudiced on the results such as quality of image, size of image (e.g.

648x480) and the parameters of recognition techniques or algorithm. In first three algorithms e.g. Row Vector, Edging and Row vector and Mean and Standard Deviation, neural network used for training and testing but there is real time classification training and testing without NN in Diagonal Sum, So its takes less time. Therefore, time counted for training phase. Given image to system for testing it's include training, testing, feature extraction and recognition of that particular image Following are comparison of processing time for all algorithms:

Algori thms	**Row Vect or (NN)**	**Edgi ng and Row vecto r (NN)**	**Mea n & S.D (NN)**	**Dia gon al Sum**
Proce ssor Speed	Intel 2.53 GHz	Intel 2.53 GHz	Intel 2.53 GHz	Intel 2.53 GHz
Test Image Size	640x 480	640x 480	640x 480	640 x48 0
Time (s)	240 sec	240 sec	180 sec	60 sec

Table 1: Processing Time (Testing)

4.14 ROTATION VARIANT

Influence of rotation in same gesture at different degree is also important role in gesture recognition process. Let consider first three methods e.g. Row vector, Edging and Row Vector and Mean & S.D. There are multiple images of different people with different degrees of angle in training database so neural network have ability of learning which can classify with different variation of gesture position. Increasing training pattern gives more effective result because I observe that neural network able to generalize better as if we increase number of gestures patterns made by different people, there by better ability to extract features of specific gesture rather than feature of same gesture which made by single person. The main motivation of neural network in pattern recognition is that once network set properly trained by learning then system produce good result even existence of incomplete pattern and noise.

In real time classification e.g. Diagonal Sum rotation influence does matter but it depends on degree of rotation. It can easily classify if degree of rotation if between 10 to 15 degree which you can see below Fig 4.15. But if the degree of rotation is more that this then it is possible that it misclassifies. Let suppose during training process if we are going to give gesture to system which originate vertically and for testing it’s on vertically see below Fig 4.15. So it is possible that diagonal sum value will change and output also misclassify. This is because when we try to recognize gesture in real time, determining problem during training when one gesture end other begins, so this is main cause of misclassification.

Fig 4.15: Degree of Rotation classification

Fig 4.16: Degree of Rotation misclassification

4.15 EXPERIMENTS AND ANALYSIS

Performed experiment shows the achieved result and estimate gesture recognition system projected in chapter 4. The experiment divided into two categories to better analyze system performance and capabilities. The more general approach to work with differently user independent system developed to interact with multi users with different kind of skin colors and hands shapes. It is very important approach to attempt for independent multi- user system. The system can be used by various users.

Two main aims for this work to detect hand and recognition of hand gesture with neural network and real classification. The first aim to detect hand with different skin tones, using explicitly defined skin region. Secondly gesture recognition with neural network and real classification by different algorithms. This system designed to test the hypothesis that detection and recognition rate would increase as:

- Hand detection with different skin tones
- More training pattern are used to train neural network
- Gesture recognition

The analysis of each experiment which mentioned above is presented here one by one according to above sequence.

4.16 EFFECT WITH DIFFERENT SKIN TONES:

Hand detection with different ethnic group experiment designed to test hypothesis that detection rate would increase. If we are able to detect skin then ultimately hand will detect. For using explicitly defined skin region classifier makes this possible to detect almost all kind of skins with different ethnic group.

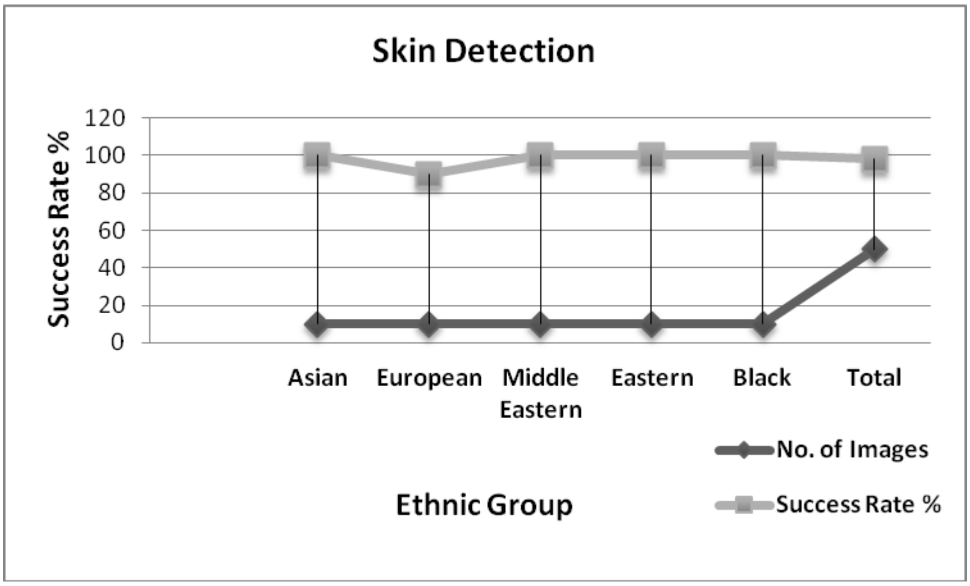


Fig 4.17: Ethnic Group Skin Detection

Figure 4.17 is a graph showing that effect of using explicitly skin classifier with different ethnic group. 10 images of each group given to this and it give almost 100% result. It has ability to detect almost all kind of skin tones. Using explicitly defined skin region makes possible and experiment have confirmed this with different skin tones.

Since it was observed that the detection accuracy varied with different skin colors and light intensity but using skin classifier make system efficient can have ability to detect different skin tones.

4.17 EFFECT OF TRAINING PATTERN:

This experimental hypothesis was to decreased misclassification as the number of training pattern increased. There are multiple images of different people with different degrees of angle in training database so neural network have ability of learning which can classify with different variation of gesture position. Increasing training pattern gives more effective results.

Initially in neural network try to train network with 30 images as 6 images of individual gesture but classification result was less and it is observed that as we increase number of pattern of different people classification result increased which we can see Figure 4.18 by increasing training set classification result improved.

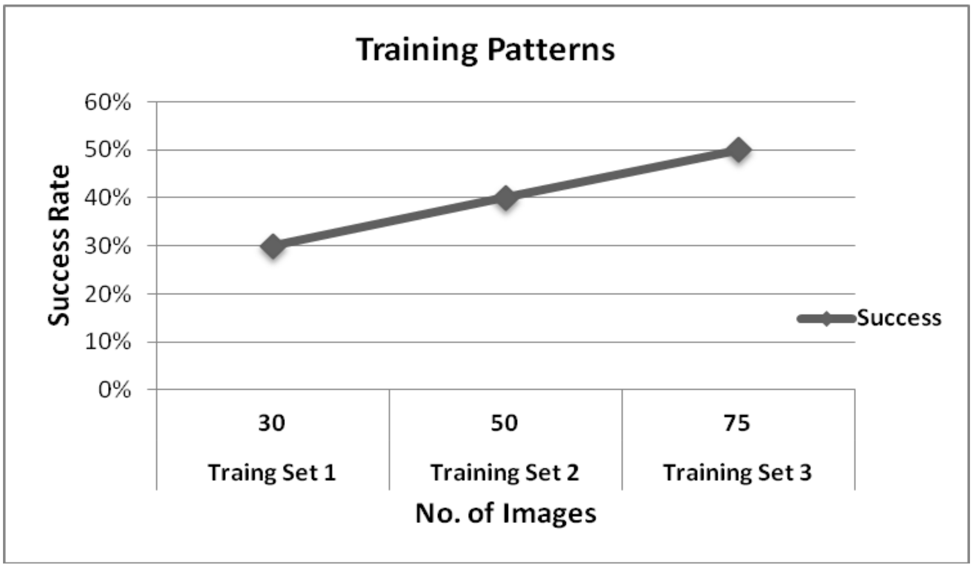


Fig 4.18 : Mix Training Pattern

This experiment shows that neural network able to generalize better as if we increase number of gestures patterns made by different people, there by better ability to extract features of specific gesture rather than feature of same gesture which made by single person. The main motivation of neural network in pattern recognition is that once network set properly trained by learning then system produce good result even existence of incomplete pattern and noise.

4.18 GESTURE RECOGNITION:

This experimental hypothesis was to recognize of gestures that user gave to system either with training with neural network and real time classification. The database to test hand gesture recognition system created 75 static images with different background. The static images size 640x480 pixels collected from camera. I grabbed gestures from different people and implement different methods of recognition with more than one user ultimately system has ability to recognize gestures of different peoples.

The accuracy percentage measured per hand gesture from different users. In Figure 5.6 prove the effectiveness of all method implemented. Classification of each gesture can be seen from following results. In following results comparison that classification percentage varies with different methods.

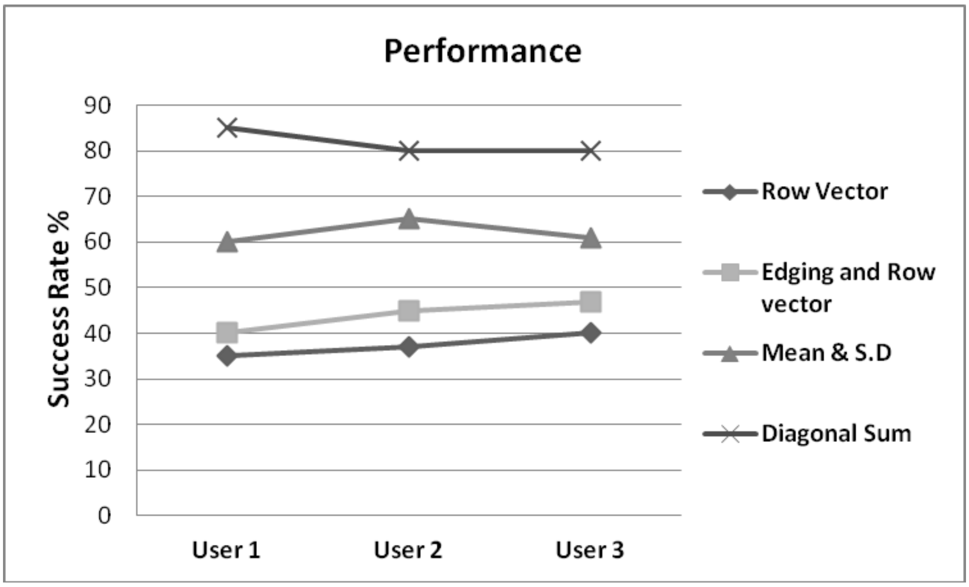


Figure 4.19 : Classification Percentage

This experiment shows that system have ability to recognize hand gestures with multiple users with different skin colors. Hand gestures classification accuracy can see while classifying between five gestures. These are not enough result to express any conclusive trend for gesture recognition experiment but most important thing is that gesture recognition was possible using neural network and real time classification.

4.19 FAILURE ANALYSIS

4.19.1 ORIENTATION/ ROTATION

There are few factors which could decrease efficiency of system specially when we talk about real time classification problem occur during training when one gesture end other begins, when user try to recognize gesture with different degree the system fail to recognize the gesture.

In different hand images the hand could have different orientation; this may affect hand recognition accuracy because when we get images from camera to testing and training distance of camera from hand and rotation of wrist makes difference, so it is essential to work with many degrees of freedom as possible in order to make recognition process realistic.

If we talk about orientation we must deal this thing in processing phase doing framing of every image by finding main axis of hand and calculate orientation and reorientation. For the rotation of wrist there are many way to deal with e.g. COG (Centre of Gravity). In processing phase we could dealt with every training and testing images by rotating clock or anti clockwise by making absolute orientation point.

4.20 CONVENTIONAL TEST STRATEGIES APPLIED

4.20.1 UNIT TESTING

The unit test focused on the internal processing logic. All statements in a module have been exercised at least once. The interface module was tested to ensure that information properly flowed into and out of the program unit under test.

4.20.2 INTEGRATION TESTING

Integration testing is a technique for constructing the software architecture and conducting tests to uncover errors with interface. The objective of testing was to crosscheck for components fully functional or not according to design. Thus, I integrated all my unit components and saw if the system worked as a whole properly or not. The information flows between the components were checked once again.

4.20.3 RECOVERY TESTING

System fails in many ways but recovery must be properly performed. For example when a person trains the system with his gestures and the system fails to do so, it gives an error message indicating that the system was not properly trained and it keeps on doing so until it gets a valid gesture (that's acceptable to the system for differentiation). When the system is properly trained, only then we can expect that it will give us accurate results.

4.20.4 SENSITIVITY TESTING

Invalid input classes that may cause instability or improper processing. It was found during sensitivity testing that if the system was once fully trained for all the gesture types, it gave accurate results, otherwise if it were just trained for a single or two gestures and then tested, it performs erroneous processing.

CHAPTER 5

PERFORMANCE EVALUATION

Hand sign recognition and finger gesture recognition can add and change training data and retrain the model.

5.1 HAND SIGN RECOGNITION TRAINING

5.1.1 LEARNING DATA COLLECTION

Press "k" to enter the mode to save key point（displayed as「MODE:Logging Key Point」）

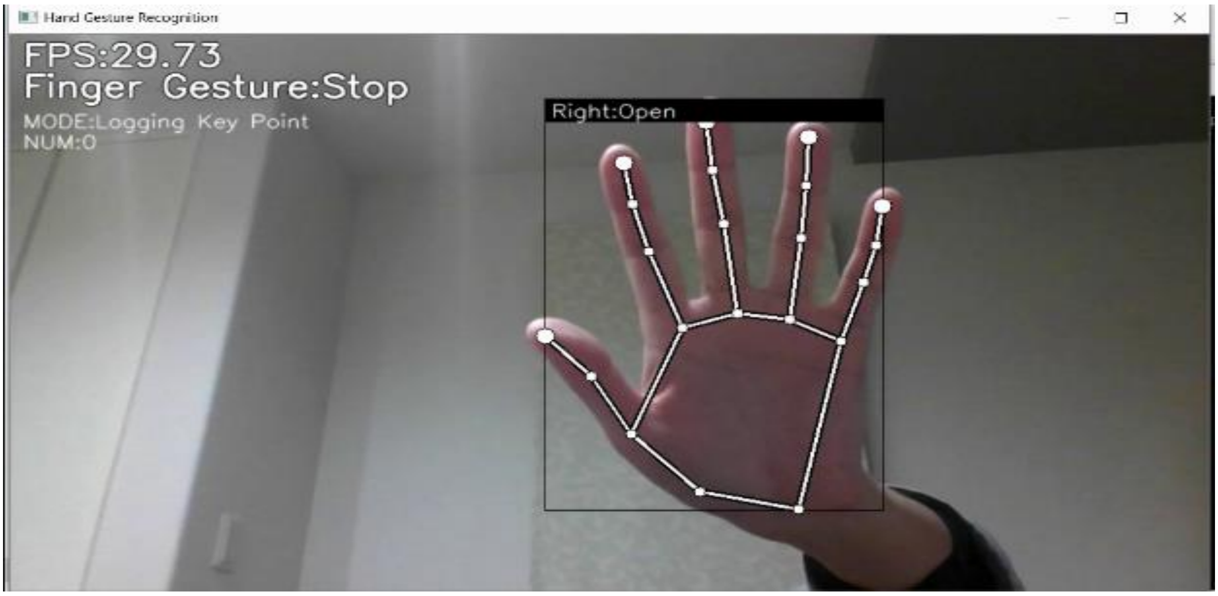


Fig 5.1: Gesture Recognition.

If you press "0" to "9", the key points will be added to "model/key point classifier/keypoint.csv" as shown below.

1st column: Pressed number (used as class ID), 2nd and subsequent columns: Key point coordinates.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1519	0	0	0	-0.21659	0.073733	-0.34101	0.253456	-0.40553	0.419355	-0.40092	0.552995	-0.28571	0.198157	-0.35945	0.479263	-0.37327	0.645161	-0.3686
1520	0	0	0	-0.2287	0.080717	-0.33632	0.255605	-0.38565	0.426009	-0.36323	0.565022	-0.26906	0.179372	-0.33184	0.452915	-0.34978	0.627803	-0.3542
1521	0	0	0	-0.16889	0.048889	-0.21778	0.217778	-0.24444	0.4	-0.24889	0.551111	-0.08	0.151111	-0.06222	0.377778	-0.02667	0.524444	0.01333
1522	0	0	0	-0.16114	0.066351	-0.22275	0.236967	-0.25118	0.417062	-0.24171	0.559242	-0.19431	0.194313	-0.2654	0.469194	-0.2891	0.649289	-0.312
1523	1	0	0	-0.3	-0.18667	-0.44667	-0.48	-0.46667	-0.76	-0.46667	-1	-0.3	-0.67333	-0.29333	-0.90667	-0.31333	-0.66667	-0.3133
1524	1	0	0	-0.32432	-0.17568	-0.5	-0.4527	-0.5473	-0.74324	-0.56757	-1	-0.41216	-0.65541	-0.39865	-0.91892	-0.38514	-0.67568	-0.3851
1525	1	0	0	-0.33803	-0.16901	-0.54225	-0.43662	-0.59859	-0.73239	-0.61972	-1	-0.4507	-0.67606	-0.43662	-0.93662	-0.41549	-0.6831	-0.4154
1526	1	0	0	-0.34286	-0.15	-0.55	-0.42857	-0.62857	-0.72857	-0.66429	-1	-0.49286	-0.66429	-0.47857	-0.93571	-0.44286	-0.67857	-0.43571

Table 5.1: Key point classifier.

The key point coordinates are the ones that have undergone the following preprocessing up to ④.

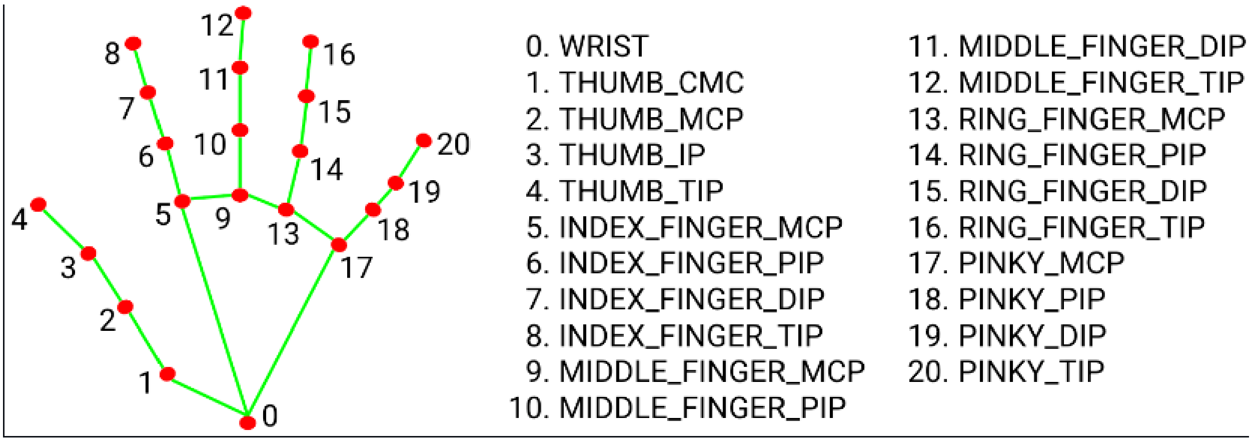


Fig 5.2 Hand Image Recognition

(Landmark coordinates)

ID : 0	ID : 1	ID : 2	ID : 3	······	ID : 17	ID : 18	ID : 19	ID : 20
[551, 465]	[485, 428]	[439, 362]	[408, 307]	······	[633, 315]	[668, 261]	[687, 225]	[702, 188]

(Convert to relative coordinates from ID:0)

ID : 0	ID : 1	ID : 2	ID : 3	······	ID : 17	ID : 18	ID : 19	ID : 20
[0, 0]	[-66, -37]	[-112, -103]	[-143, -158]	······	[82, -150]	[117, -204]	[136, -240]	[151, -277]

(Flatten to a one-dimensional array)

ID : 0		ID : 1		ID : 2		ID : 3		······	ID : 17		ID : 18		ID : 19		ID : 20	
0	0	-66	-37	-112	-103	-143	-158	······	82	-150	117	-204	136	-240	151	-277

(Normalized to the maximum value(absolute value))

ID : 0		ID : 1		ID : 2		ID : 3		······	ID : 17		ID : 18		ID : 19		ID : 20	
0	0	-0.24	-0.13	-0.4	-0.37	-0.52	-0.57	······	0.296	-0.54	0.422	-0.74	0.491	-0.87	0.545	-1

Table 5.2: Dataset Training

In the initial state, three types of learning data are included: open hand (class ID: 0), close hand (class ID: 1), and pointing (class ID: 2).

If necessary, add 3 or later, or delete the existing data of CSV to prepare the training data.

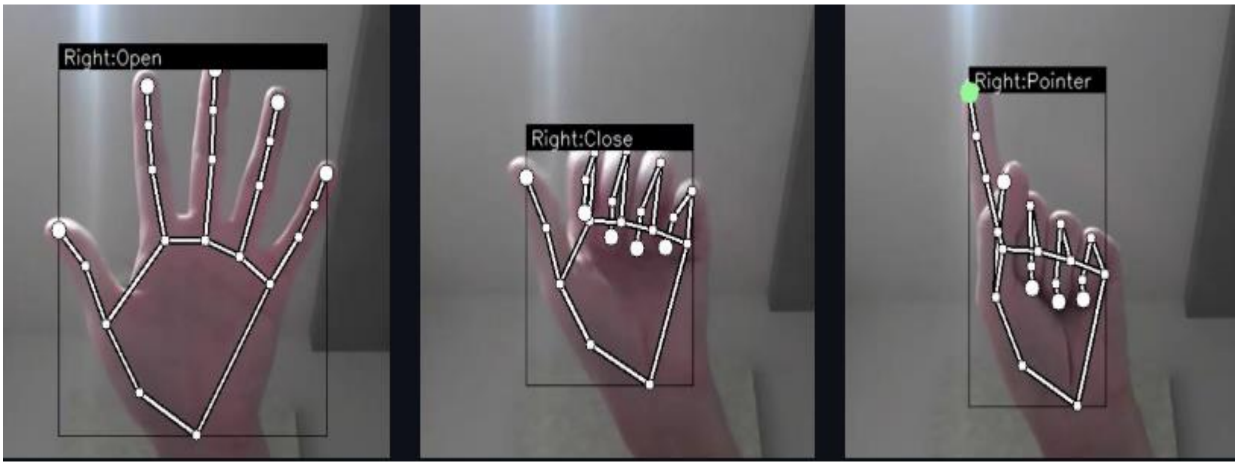


Fig 5.3: Three types of learning data.

5.1.2 MODEL TRAINING

Open "keypoint classification.ipynb" in Jupyter Notebook and execute from top to bottom. To change the number of training data classes, change the value of "NUM_CLASSES = 4" and modify the label of "model/keypoint_classifier/keypoint_classifier_label.csv" as appropriate.

X. Model structure

The image of the model prepared in "keypoint_classification.ipynb" is as follows.

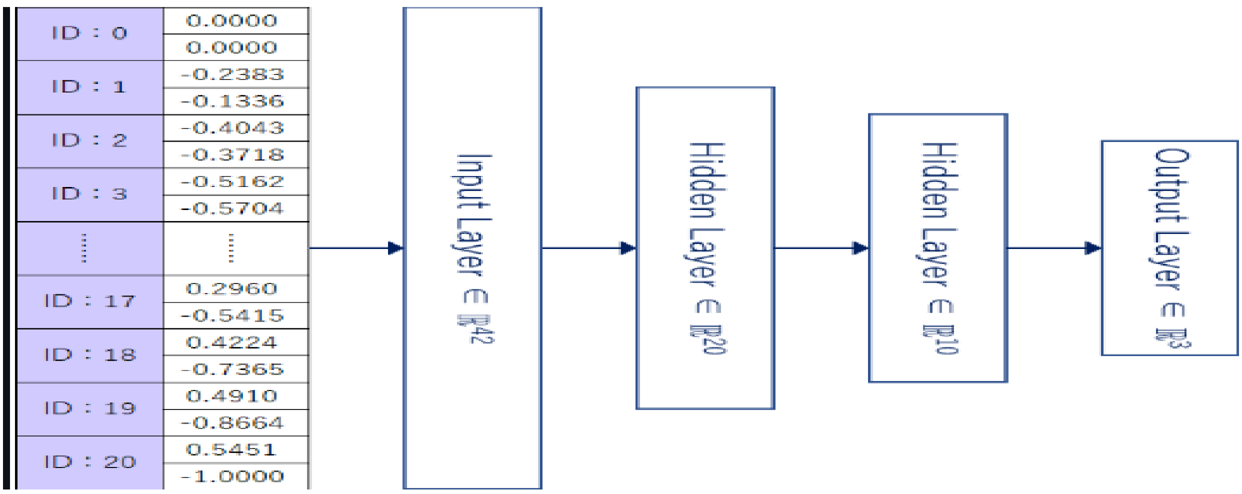


Fig 5.4: Model structure

5.2 FINGER GESTURE RECOGNITION TRAINING

5.2.1 LEARNING DATA COLLECTION

Press "h" to enter the mode to save the history of fingertip coordinates (displayed as "MODE: Logging Point History").

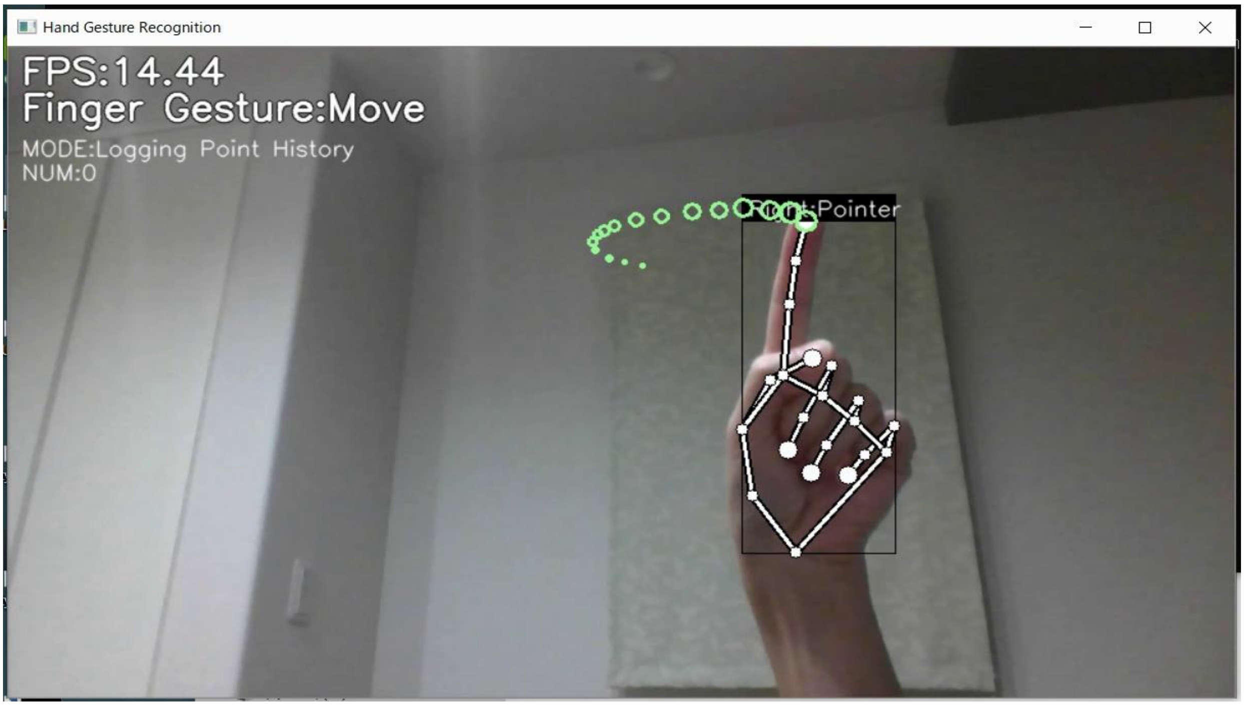


Fig 5.5: Learning data collection

1st column: Pressed number (used as class ID), 2nd and subsequent columns: Coordinate history

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1478	0	0	0	0.001042	0	0	-0.00185	0	-0.0037	0	-0.0037	0	-0.0037	0	-0.0037	0	-0.0037	0
1479	0	0	0	-0.00104	-0.00185	-0.00104	-0.0037	-0.00104	-0.0037	-0.00104	-0.0037	-0.00104	-0.0037	-0.00104	-0.0037	-0.00104	-0.00556	-0.00104
1480	0	0	0	0	-0.00185	0	-0.00185	0	-0.00185	0	-0.00185	0	-0.00185	0	-0.0037	0	-0.0037	0
1481	0	0	0	0	0	0	0	0	0	0	0	0	-0.00185	0	-0.00185	0	-0.00185	0
1482	1	0	0	-0.00313	-0.02778	0.002083	-0.05741	0.010417	-0.08148	0.020833	-0.09444	0.0375	-0.10741	0.054167	-0.10556	0.073958	-0.09444	0.088542
1483	1	0	0	-0.01354	-0.02037	-0.02292	-0.04815	-0.02292	-0.08519	-0.01875	-0.11481	-0.00938	-0.13704	0.002083	-0.15741	0.017708	-0.17037	0.034375
1484	1	0	0	-0.00938	-0.02778	-0.00938	-0.06481	-0.00521	-0.09444	0.004167	-0.11667	0.015625	-0.13704	0.03125	-0.15	0.047917	-0.1537	0.066667
1485	1	0	0	0	-0.03704	0.004167	-0.06667	0.013542	-0.08889	0.025	-0.10926	0.040625	-0.12222	0.057292	-0.12593	0.076042	-0.12037	0.09375

Table 5.4: Coordinate History

The key point coordinates are the ones that have undergone the following preprocessing up to ④.

<table>
<tr><td colspan="14">(Time series coordinates)</td></tr>
<tr><td colspan="2">T-15</td><td colspan="2">T-14</td><td colspan="2">T-13</td><td>……</td><td colspan="2">T-2</td><td colspan="2">T-1</td><td colspan="2">T</td></tr>
<tr><td colspan="2">[550, 165]</td><td colspan="2">[526, 176]</td><td colspan="2">[509, 188]</td><td>……</td><td colspan="2">[644, 219]</td><td colspan="2">[644, 196]</td><td colspan="2">[642, 178]</td></tr>
<tr><td colspan="14">(Convert to relative coordinates from [T-15])</td></tr>
<tr><td colspan="2">T-15</td><td colspan="2">T-14</td><td colspan="2">T-13</td><td>……</td><td colspan="2">T-2</td><td colspan="2">T-1</td><td colspan="2">T</td></tr>
<tr><td colspan="2">[0, 0]</td><td colspan="2">[-24, 11]</td><td colspan="2">[-17, 12]</td><td>……</td><td colspan="2">[5, -16]</td><td colspan="2">[0, -23]</td><td colspan="2">[-2, -18]</td></tr>
<tr><td colspan="14">(Normalized to fit screen width and height)</td></tr>
<tr><td colspan="14">T-14は[-24/960, 11/540] (Width of 960 and height of 540, T-14 is [-24/960, 11/540])</td></tr>
<tr><td colspan="2">T-15</td><td colspan="2">T-14</td><td colspan="2">T-13</td><td>……</td><td colspan="2">T-2</td><td colspan="2">T-1</td><td colspan="2">T</td></tr>
<tr><td colspan="2">[0.0, 0.0]</td><td colspan="2">[-0.025, 0.0204]</td><td colspan="2">[-0.0427, 0.0426]</td><td>……</td><td colspan="2">[0.0979, 0.1]</td><td colspan="2">[0.0979, 0.0574]</td><td colspan="2">[0.0958, 0.024]</td></tr>
<tr><td colspan="14">(Flatten to a one-dimensional array)</td></tr>
<tr><td colspan="2">T-15</td><td colspan="2">T-14</td><td colspan="2">T-13</td><td>……</td><td colspan="2">T-2</td><td colspan="2">T-1</td><td colspan="2">T</td></tr>
<tr><td>0.0000</td><td>0.0000</td><td>-0.0250</td><td>0.0204</td><td>-0.0427</td><td>0.0426</td><td>……</td><td>0.0979</td><td>0.1000</td><td>0.0979</td><td>0.0574</td><td>0.0958</td><td>0.0241</td></tr>
</table>

Table 5.5: Key point coordinates.

In the initial state, 4 types of learning data are included: stationary (class ID: 0), clockwise (class ID: 1), counterclockwise (class ID: 2), and moving (class ID: 4).
If necessary, add 5 or later, or delete the existing data of csv to prepare the training data.

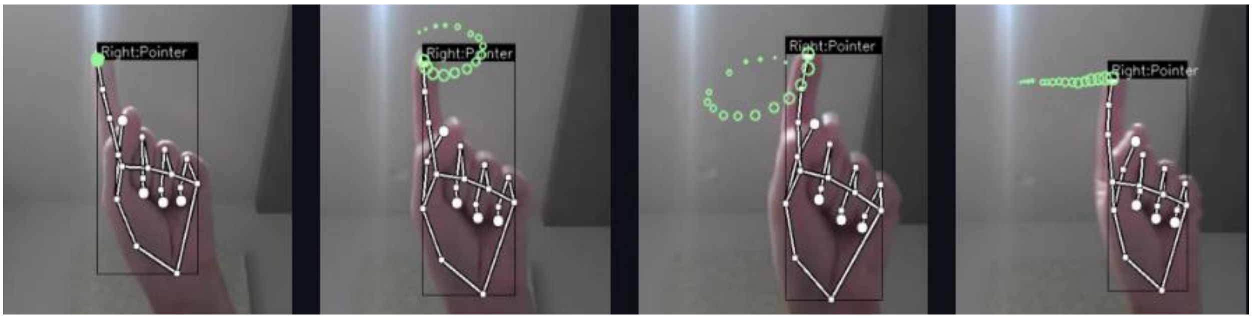


Fig 5.6: Four types of learning data

5.2.2 MODEL TRAINING

Open "point_history_classification.ipynb" in Jupyter Notebook and execute from top to bottom. To change the number of training data classes, change the value of "NUM_CLASSES = 4" and modify the label of "model/point_history_classifier/point_history_classifier_label.csv" as appropriate.

X. Model structure

The image of the model prepared in "point_history_classification.ipynb" is as follows.

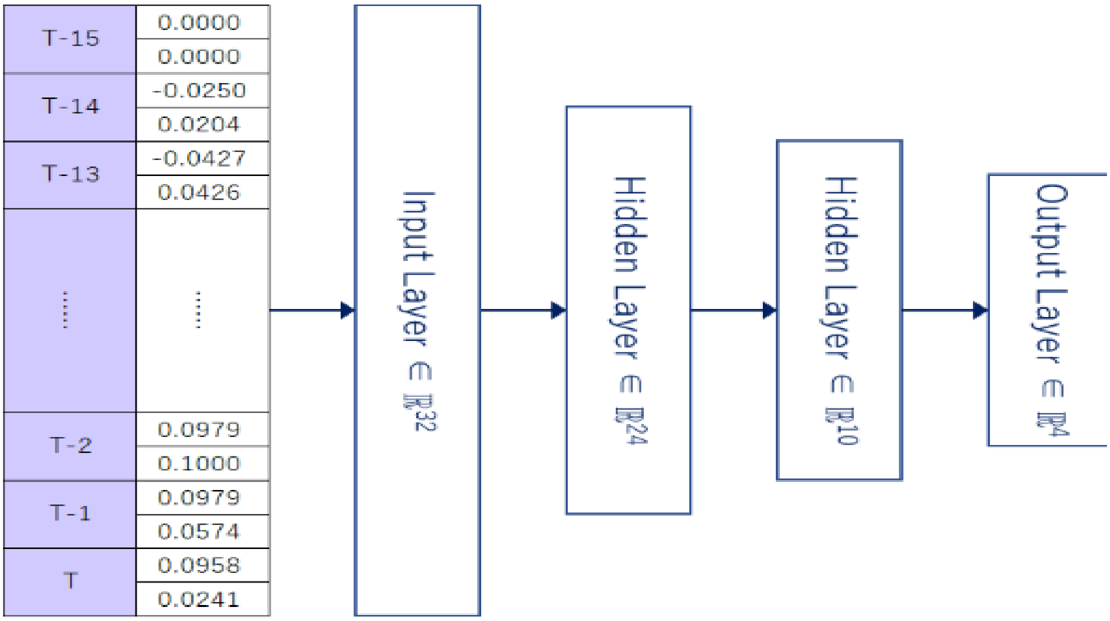


Fig 5.7: Point_history_classification.

Please change "use_lstm = False" to "True" when using (tf-nightly required (as of 2020/12/16))

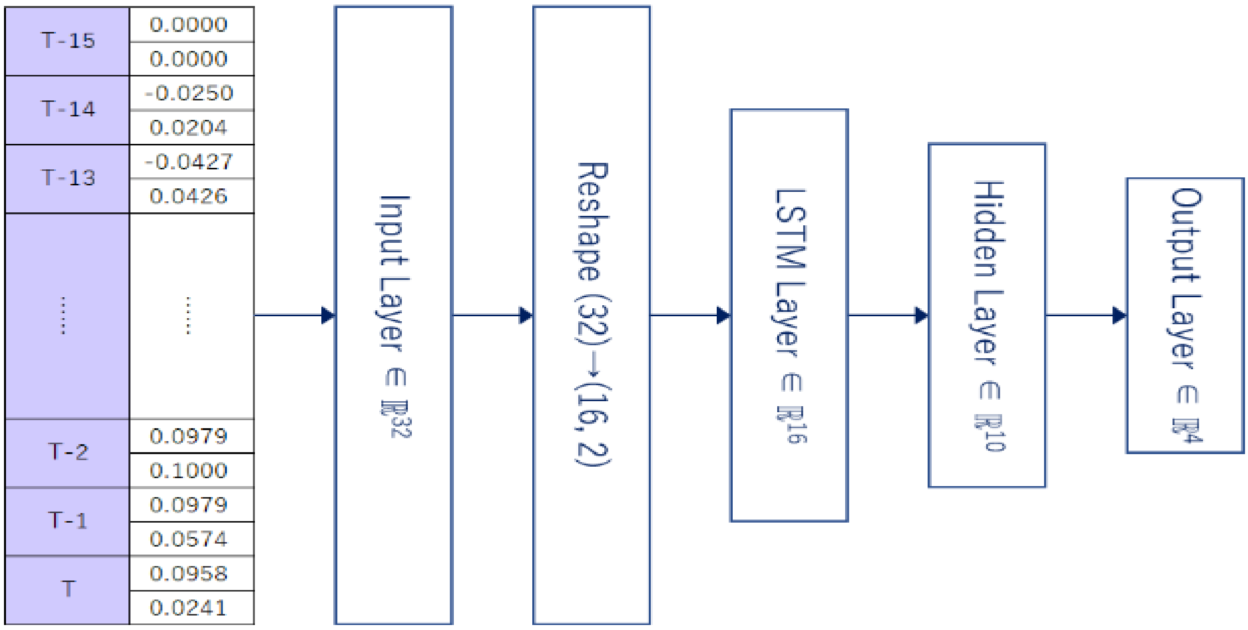


Fig 5.8: Point_history_classification.

5.3 CONFUSION MATRIX

After we were satisfied with the training data set, we conducted a constrained, lab-based evaluation of our gesture detection classifier. 3.1 Pilot Study Seventeen undergraduate and graduate student volunteers tested this hand tracking and gesture recognition system. Each participant was instructed in each of the gestures and provided a brief demo of the system. When they were ready to start, the computer randomly selected a gesture name and showed them on the screen one at a time. Each participant was instructed to generate 27-30 specific gestures (each lasting 7 seconds). 3.2 Initial Results We compared the gestures that the participants were asked to produce with the gesture that was detected by the classifier. We want to note that we are omitting the RM and LM (hand on mat) from this analysis. While the overall recognition rate for these gestures was fairly high (RM = 88% and LM = 90%), several of the participants placed their hand on the table during the study and kept one hand on the table while doing other gestures, which made the detection of this gesture function differently from the others.

The RM and LM gestures are only necessary for the “hold hands” gesture and this gesture is only relevant when combining Squeeze Bands with the Share Table system (versus standard video chat). Given that the share Table system currently provides accurate 20-point multi-touch detection for locating a hand on the projection area, the classifier for this particular gesture may be redundant. The recognition rates obtained for the six retained gestures are shown in the form of confusion matrix in Figure 6.1. The overall recognition rate for these six gestures is 89%. While the accuracy of detecting handshakes with right hand and patting shoulder with left hand are relatively lower than other gestures, the general accuracy of the recognition was fairly high. This follow-up study presents the preliminary feasibility of automatically detecting gestures directed to a video chat partner for the purpose of triggering a hap tic device. Three factors may increase or decrease this accuracy in the field. First, gestures enacted naturalistically may have lower accuracy than ones enacted in response to an instruction because there may be contextual factors influencing the specifics of how a user moves. Additionally, in the field the system will have to differentiate not just between the six types of gestures but also whether any gesture was attempted at all. However, there are also two factors that may also increase accuracy.

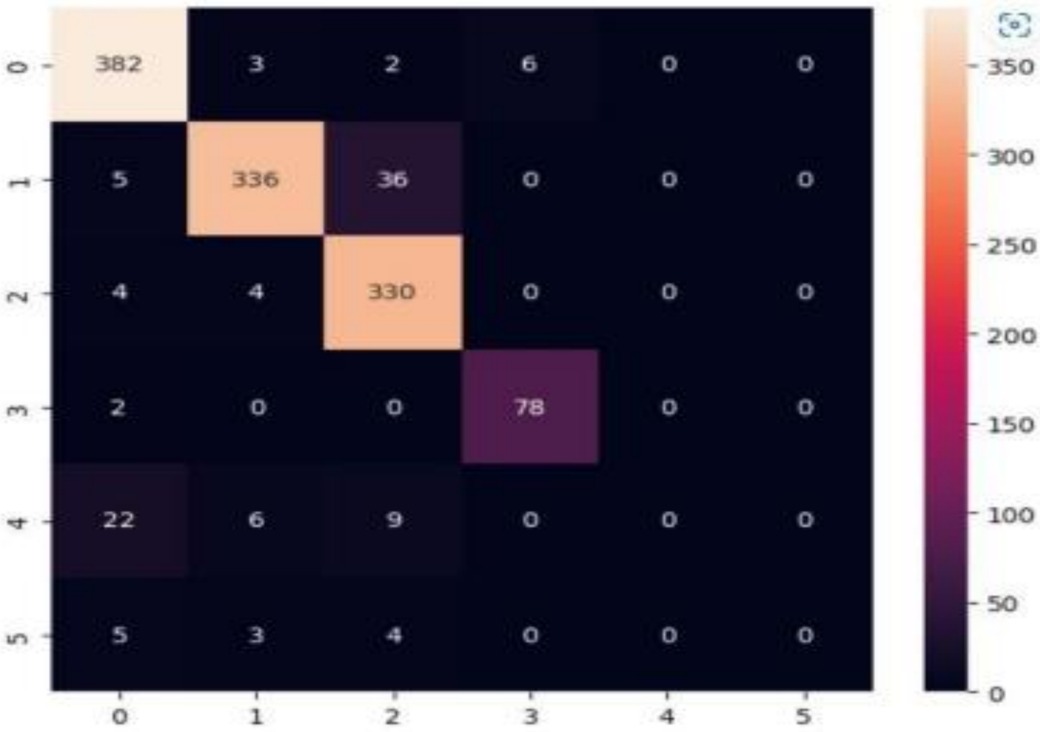


Fig: 5.9: Confusion matrix.

Classification Report

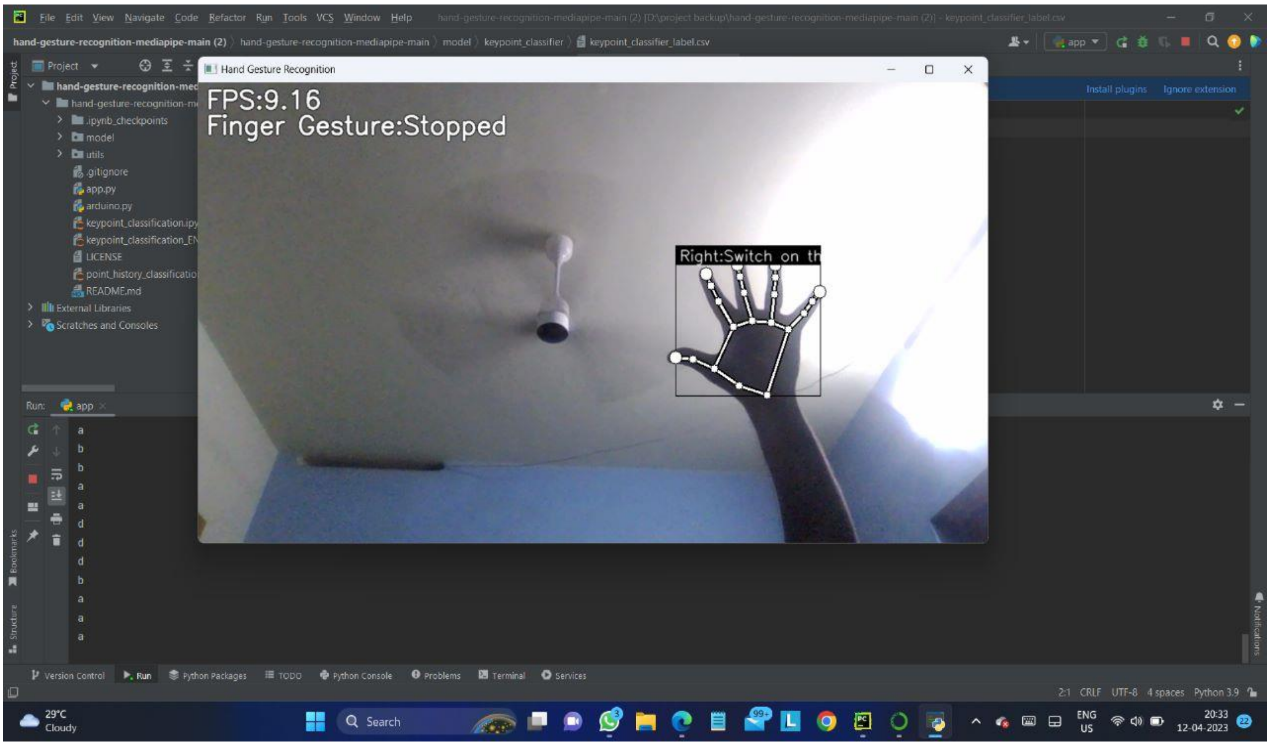
	precision	recall	f1-score	support
0	0.91	0.97	0.94	393
1	0.95	0.89	0.92	377
2	0.87	0.98	0.92	338
3	0.93	0.97	0.95	80
4	0.00	0.00	0.00	37
5	0.00	0.00	0.00	12
accuracy			0.91	1237
macro avg	0.61	0.64	0.62	1237
weighted avg	0.88	0.91	0.89	1237

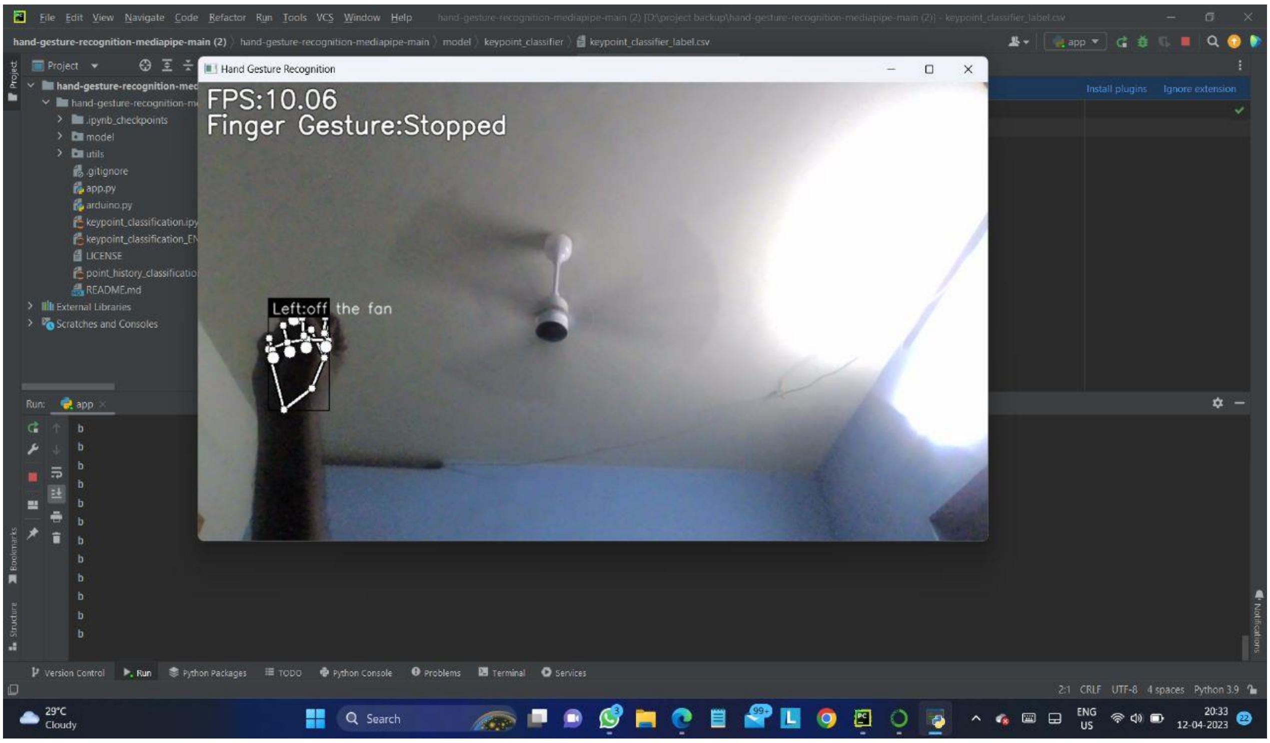
Table 5.6: Classification Report.

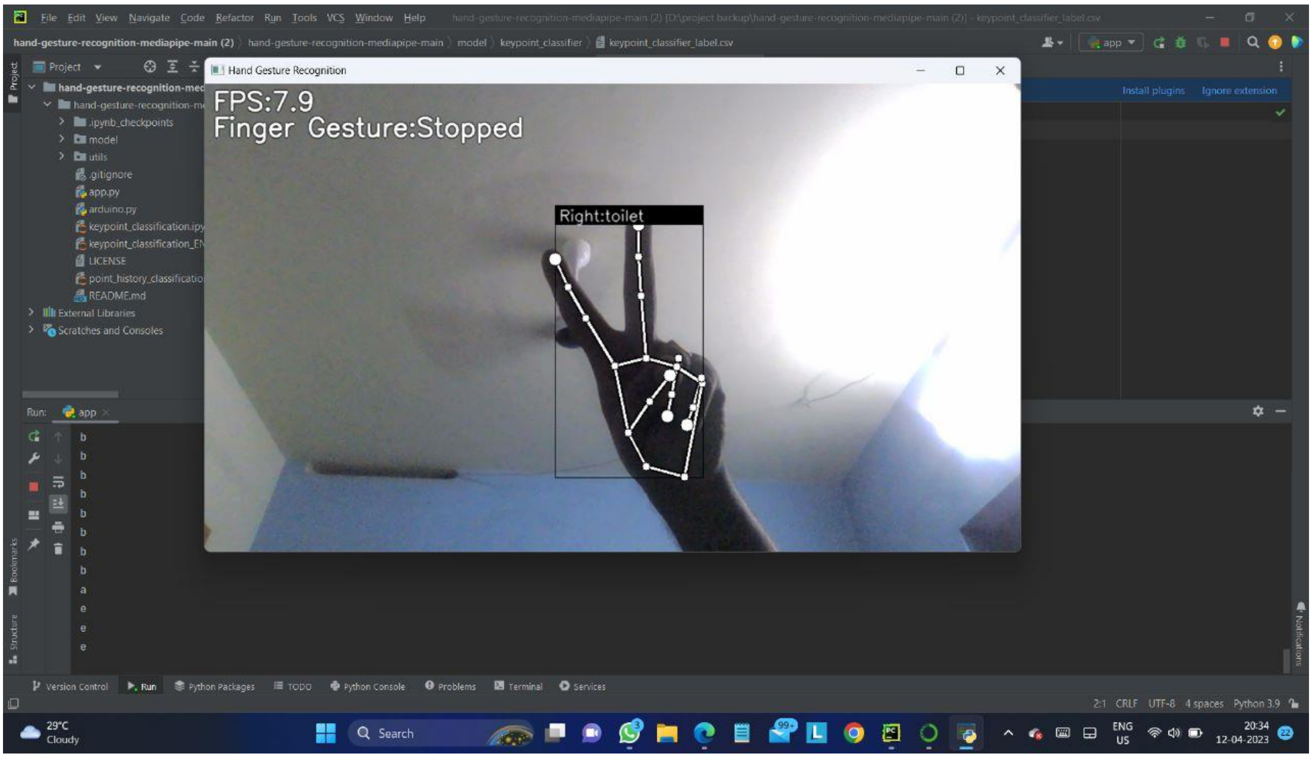
First, we could incorporate a brief user-specific training session when installing the Squeeze Bands system. A user-specific classifier is likely to have higher accuracy than a general classifier employed by our system. Second, two-thirds of the gestures (L5, R5, LH, RH) classified are dyadic in nature. For example, a handshake should only be classified as a handshake if the system is reasonably confident that both participants are attempting the same gesture. Thus, high confidence for one of the participants may make up for low confidence for the other participant in selecting the correct gesture. Overall, despite the initial promise shown by our preliminary gesture detection system, significant future work is necessary to demonstrate its effectiveness in the field.

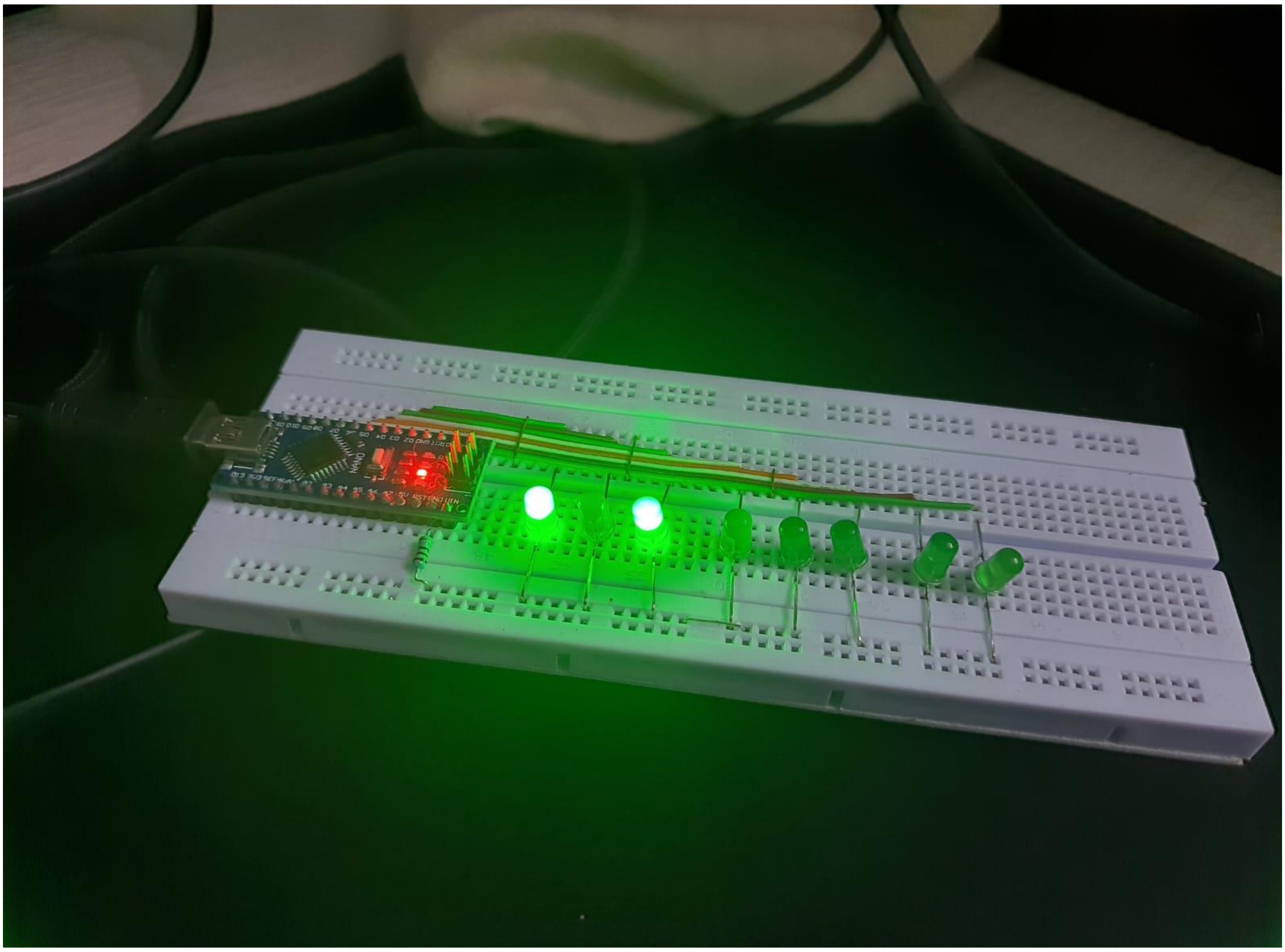
CHAPTER 6

OUTPUT






Hand Gesture Recognition
FPS:7.9
Finger Gesture:Stopped
Right:toilet



CHAPTER 7

CONCLUSION

This chapter summarizes my work at every stage of the project. At the time I started my thesis, I had a brief idea of how I will bring it from a topic on the paper to a real product. Due to knowledge of Computer Vision and Biometric subjects I had background in the image-processing field but not at expert level but my constant effort helped me to go through and succeed eventually. As required in every project, research is of utmost importance. So, I spent the pretty much time in going through the background literature. I looked at various approaches of doing my thesis and developed four different methods: Row vector algorithm, Edging and row vector passing algorithm, Mean and standard deviation of edged image and Diagonal sum algorithm. Each of these algorithms was tried with neural networks and have higher performance rate in the ascending order respectively. The first limitation that was discovered in all the algorithms used with neural networks was that their performance depended on the amount of training dataset provided. The system worked efficiently after being trained by a larger dataset as compared to a smaller dataset.

The Row vector algorithm used initially was a very vague approach adopted for classification as it was found through experiments that the row vectors of two different images could happen to be the same.

In the edging and row vector-passing algorithm, the edging parameter was introduced in addition to the row vector to improve the gesture classification accuracy but it was found that due to self- shadowing effect found in edges, the detection rate was not sufficiently improved. The next parameters tried for classification were mean and standard deviation. They also failed to give satisfactory results (i.e. above 60%) but still they were among the best parameters used for detection with neural networks.

Due to the unusual behaviour of neural network with all the mentioned parameters, the diagonal sum algorithm was finally implemented in real time. The system was tested with 60 pictures and the detection rate was found to be 86%. The strengths and weaknesses of gesture recognition using diagonal sum have been presented and discussed. With the implemented

CHAPTER 8

FUTURE SCOPE

The system could also be made smart to be trained for only one or two gestures rather than all and then made ready for testing. This will require only a few changes in the current interface code, which were not performed due to the shortage of time.

One time training constraint for real time system can be removed if the algorithm is made efficient to work with all skin types and light conditions which seems impossible by now altogether. Framing with COG (Centre of gravity) to control orientation factor could make this system more perfect for real application

REFERENCES

[1] J.Jenkinwinston (96207106036), M.Maria Gnanam (96207106056), R.Ramasamy (96207106306), Anna University of Technology, Tirunelveli: Hand Gesture Recognition system Using Haar Wavelet.

[2] Laura Dipietro, Angelo M. Sabatini, Senior Member, IEEE, and Paolo Dario, Fellow, IEEE, A Survey of Glove-Based Systems and Their Applications.

[3] Kay M. Stanney HANDBOOK OF VIRTUAL ENVIRONMENTS Design, Implementation, and Applications, Gesture Recognition Chapter #10 by MatthewTurk

[4] Daniel Thalman, Gesture Recognition Motion Capture, Motion Retargeting, and Action Recognition

[5] Hatice Gunes, Massimo Piccardi, Tony Ja, 2007, Research School of Information Sciences and Engineering Australian National University Face and Body Gesture Recognition for a Vision-Based Multimodal Analyzer

[6] J. Heinzm ann and A. Zelinsky Robust Real - Time Face Tracking and Gesture Recognition

[7] Vladimir Vezhnevets, Vassili Sazonov, Alla Andreeva, Moscow State University A Survey on Pixel-Based Skin Color Detection Techniques,

[8] TEY YI CHI, Universiti Teknologi Malaysi, FUZZY SKIN DETECTION
Robyn Ball and Philippe Tissot, Texas A&M University, Demonstration of Artificial Neural Network in Matlab

[9] Howard Demuth, Mark Beale, Neural Network Toolbox

[10] Ari Y. Benbasat and Joseph A. Paradiso in MIT Media Laboratory, Cambridge An Inertial Measurement Framework for Gesture Recognition and Applications

[11] Peggy Gerardin, Color Imaging Course Winter Semester 2002-2003, Color Image Segmentation

[12] Michael Nielsen, Moritz Störring, Thomas B. Moeslund, and Erik Granum, March 2003, A procedure for developing intuitive and ergonomic gesture interfaces for man-machine interaction.