

GESTURE AUTOMATION SYSTEM FOR VARIOUS APPLICATIONS

*in partial fulfillment of the requirements for the award of the degree
of Bachelor of Engineering in*

COMPUTER ENGINEERING

Submitted by

NAME OF THE CANDIDATES

AKANKSHA RAUT

NIDHI DONTULWAR

HIMANSHU Bhole

HRISHIKESH DHOLE

Under the guidance of

Prof. PALLAVI WANKHEDE

Assistant Professor
Project Guide
Computer Engineering

Academic Year 2020-21
Department of Computer Engineering



ST. VINCENT PALLOTTI COLLEGE OF ENGINEERING AND TECHNOLOGY

Wardha Road, Gavsi Manapur, Nagpur

ST. VINCENT PALLOTTI COLLEGE OF ENGINEERING AND TECHNOLOGY

Wardha Road, Gavsi Manapur, Nagpur

Department of Computer Engineering

CERTIFICATE

Certified that this project report “**GESTURE AUTOMATION SYSTEM FOR VARIOUS APPLICATIONS**” is the bonafide work of “**AKANKSHA RAUT, NIDHI DONTULWAR, HIMANSHU Bhole, HRISHIKESH DHOLE**” who carried out the project work under my supervision in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in **COMPUTER ENGINEERING** of RASHTRASANT TUKADOJI MAHARAJ NAGPUR UNIVERSITY, NAGPUR

Prof. S. M. Wanjari

Assistant Professor

Head of the Department

Computer Engineering

Prof. Pallavi Wankhede

Assistant Professor

Project Guide

Computer Engineering

PRINCIPAL

INDUSTRY CERTIFICATE



TANTRANSH SOLUTIONS

Tantransh Solutions

Plot no. 194, Near Kasturba
Bhavan, Bajaj Nagar, Nagpur 40010

9890282949

admin@tantranshsolutions.com

Ref: TS/INT/2021/06-01

Date: 19-06-2021

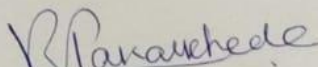
TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Akanksha Raut, Nidhi Dontulwar, Himanshu Bhole, Hrishikesh Dhole** Computer Engineering, final year students from **St. Vincent Pallotti College of Engineering and Technology, Nagpur** have completed their project successfully in our organization during the period **8th September 2020 to 10th June 2021**.

They have worked on the project titled "**Gesture automation system for various applications**".

During their work tenure with us, they were found hard-working, honest and completely dedicated to their work.

We wish them all the best for their future endeavors.


For Tantransh Solutions

Project Manager



www.tantranshsolutions.com

ACKNOWLEDGEMENT

Our major project “GESTURE AUTOMATION SYSTEM FOR VARIOUS APPLICATIONS” was carried under the guidance of Prof. Pallavi Wankhede. We wish to express our sincere gratitude to her, who convincingly guided and encouraged us to be professional and do the right thing even when the road got tough. Without her persistent help, the goal of this project would not have been realized.

We are grateful to “TANTRANSH SOLUTIONS” for constant support and Mr. Vinod Takarkhede for giving us this opportunity so that we could contribute our skills for something good.

We would also like to thank Prof. S. M. Wanjari Head of the department of Computer Engineering and all the faculty members who kept us going on and this work would not have been possible without their inputs.

We are also grateful to our Management of the College, Principal Dr. Surendra Gole, Vice principal Prof. R.B Gowardhan for their constant support and providing us with this platform.

CONTENTS

CHAPTER NO.	TITLE	PAGE
	ABSTRACT	6
1.	INTRODUCTION	7
	1.1 GENERAL	8
2.	LITERATURE REVIEW	9
	2.1 RELATED WORK	10
	2.2 PAPER REVIEW OUTCOME	10
	2.3 LIBRARIES	11
	2.3.1 KERAS	11
	2.3.2 ANACONDA	11
	2.3.3 JUPYTER	11
	2.3.4 TENSORFLOW	12
	2.4 ALGORITHMS AND METHODS	13
	2.4.1 DATA AUGUMENTATION	13
	2.4.2 CONVOLUTION NEURAL NETWORK	13
	2.4.3 RESIDUAL NEURAL NETWORK	13
3.	PROJECT PLANNING AND SCHEDULING	14
4.	SYSTEM DESIGN	16
	4.1 SYSTEM ARCHITECTURE	17
5.	MODULES IN THE PROJECT	19
6.	REQUIREMENTS	30
7.	IMPLEMENTATION AND TESTING	32
	6.1 IMPLEMENTATION	33
	6.2 TESTING	36
8.	CONCLUSION AND FUTURE SCOPE	37
	REFEERNCES	39
	PROJECT TEAM MEMBER INFORMATION	40

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO
3	Time-line Chart	14
4.1	System architecture for gesture-based media control system	17
5.1	Some common image transformation applied for data augmentation	20
5.2	Array of RGB Matrix	21
5.3	Some Common Filters Strides	22
5.4	Stride of 2 pixels	22
5.5	RELU Operation	23
5.6	Complete CNN Architecture	23
5.7	Architecture of Residual Network for Object Photo Classification	25
5.8	Epoch	26
5.9	Accuracy and Loss Graph	27
7.1.1	Gesture Identification: Drumming Fingers	33
7.1.2	Gesture Identification: Stop Sign	33
7.1.3	Gesture Identification: Swiping Up	34
7.1.4	Gesture Identification: Swiping Down	34
7.1.5	Gesture Identification: Thumbs Up	35
7.1.6	Gesture Identification: Swiping Left	35
7.1.7	Gesture Identification: Swiping Right	36

ABSTRACT

Hand gestures are the most common forms of communication and have great importance in our world. They can help in building safe and comfortable user interfaces for a multitude of applications.

Human gesture is a method of non - verbal connection medium and can give the most instinctive, originative and normal approach to associate with the computers. Our fundamental objective is to make the connection among human and PC as normal as the collaboration between people.

Various computer vision algorithms have employed color and depth camera for hand gesture recognition, but robust classification of gestures from different subjects is still challenging.

The objective of this report is to recognize the static hand gesture images (i.e. frames) based on shapes and orientations of hand which is extracted from input video stream recorded in stable lighting and simple background conditions and execute the gesture associate with it. We can use this vision based recognized gestures to control multimedia applications.

CHAPTER 1

INTRODUCTION

INTRODUCTION

There has been a great emphasis lately on Human Computer-Interaction (HCI) research to create easy-to-use interfaces by directly employing natural communication and manipulation skills of humans. Among different human body parts, the hand is the most effective general-purpose interaction tool, due to its dexterity. The word gesture is used for many different phenomena involving human movement, especially of the hands and arms only some of these are interactive or communicative.

The main purpose of gesture recognition is to identify a particular human gesture and convey information to the computer. The gesture recognition technique is an important technology for friendly human-computer interaction and has received a lot of attention in recent years. This kind of applications requires restricted background, set of gesture commands and a camera for image capturing. We have considered single handed gestures and their directional motion defines a gesture for the application. In this application image acquisition is done using a web cam.

Some frequently used gestures in Windows media players are used and thus applying controls to the Windows media player using the predefined functions.

The finger count is very user-friendly method and is easier training to the module.

CHAPTER 2

REVIEW OF LITERATURE AND ANALYSIS

REVIEW OF LITERATURE AND ANALYSIS

Literature study is an important part of a project. It is aimed to provide a required knowledge base, gives a possibility to get familiar with available materials and solutions, also it helps to generate new ideas and to make a proper design choice. Another aspect of interest is how fast this data can be processed, since it is crucial for our system to perform calculations under real-time constraints.

2.1 Related works and Scientific Paper review

To effectively reach a more informed decision about potential approaches and feasible implementation design architecture, it is important to go through scientific research papers on some categorized directions.

In 2014, N. Krishna Chaitanya and Janardhan Rao presents

“Controlling of windows media player application using hand gesture recognition”, this system uses various hand gestures as input to operate the windows media player application. This system uses single hand gestures and its directional motion which defines a gesture for the above- mentioned application. In this system decision tree has been used for classification. [1]

In 2015, Neha S. Rokade, Harsha R. Jadhav, Sabiha A.

Pathan, Uma Annamalai “Controlling Multimedia Applications Using Hand Gesture Recognition”. In some previously implemented system, the costly 3- D sensors like Kinect are used for gesture recognition. To reduce the cost, we are using simple web camera of our laptops. The separate training set is not required to recognize the gestures, so there is no need to maintain any database for storing the frames of images. Our focus in future is on the extending the gestures for the gaming and voice for mute. In this project they use HSV scale image method for detecting and scaling the image and applying a thresholding image value and filtered the image.

2.2 Papers Review Outcome

The results and ideas presented in papers above are interesting and worth to consider for the future investigation. It was indeed a learning experience after reviewing those related scientific papers and trying out some of the implementation really helped us understand what works better. We realized that some papers are better than the others in certain aspects while others are better in other aspects. After a scrutiny to meet our goal, our choice of interests was in using Neural Networks to recognize the different hand gestures for the control of media player.

2.3 Libraries and Software Review

During software survey preferences were given to the open- source and GPL licensed software to meet low budget requirements of the project. Many online and open-source algorithms we must go through to develop our own and make it work. There have been a tremendous work-flow of Gesture recognition, vision-based hand gesture recognition, Gesture based terminal application literally flowing through the world. Our research on scientific papers gained some theoretical idea. The most interesting open-source libraries and software packages with related description are presented below.

2.3.1 Keras

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code.

2.3.2 Anaconda

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment.

2.3.3 Jupyter Notebook

Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document.

2.3.4 TensorFlow:

TensorFlow is an open source artificial intelligence library, using data flow graphs to build models. It allows developers to create large-scale neural networks with many layers. TensorFlow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation

2.4.1 Algorithms and Methods

2.4.2 Data Augmentation

Data augmentation in data analysis are techniques used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data.

It acts as a Regularizer and helps reduce overfitting when training a machine learning Model. It is closely related to oversampling in data analysis.

2.4.2 Convolution Neural Network (CNN)

The Convolution Neural Network plays an important role in processing of any picture be it image perception and classifications. Some range of fields where we use the CNN can be named as detection, recognition of faces etc.

The idea behind working of CNN is that it takes an input image compute it and identify it under certain divisions, for example Dog, cat, tiger, lion etc.

Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension).

2.4.3 Residual Neural Network (Resnet 101)

A residual neural network (ResNet) is an artificial neural network (ANN) of a sort that expands on develops known from pyramidal cells in the cerebral cortex.

Residual neural networks do this using skip associations, or easy routes to hop over a few layers. Typical ResNet models are implemented with double- or triple- layer skips that contain nonlinearities (ReLU) and batch normalization in between.

CHAPTER 3

PROJECT PLANNING AND SCHEDULING



Fig. 3 Time-line Chart

The project work started on 8th of September 2020. Which started with requirement gathering and analysis discussion with the Project guide and Industry mentor.

PHASE I: Requirement gathering and analysis.

PHASE II: System architecture and modules distribution

PHASE III: Data preparation and dataset downloading

PHASE IV: Training and classification

PHASE V: Automation and testing

The month of April was the Buffer period as the extra time to the project to keep it on tracks and to manage any unforeseen situations.

CHAPTER 4

SYSTEM DESIGN

SYSTEM DESIGN

In this project we have used different image pre-processing techniques, feature extraction and classification tool for recognizing the gesture in real time and give the appropriate command to the Windows Media Player

4.1 System architecture:

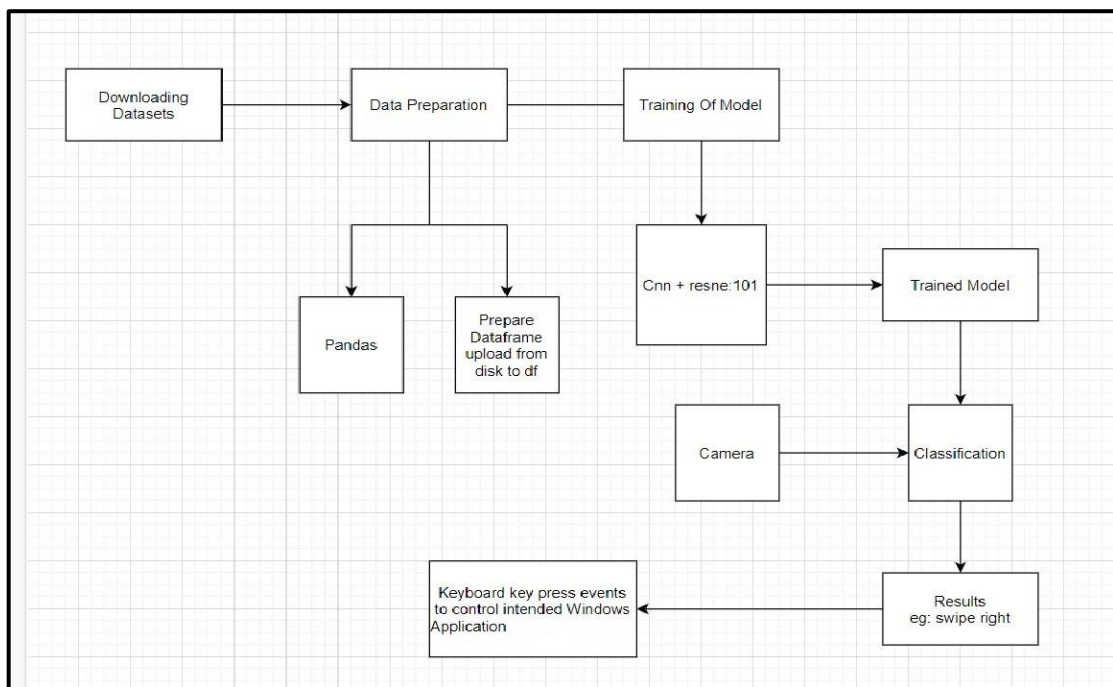


Fig. 4.1 System architecture for gesture automation system.

The above phases are described below:

Data Preparation:

The first module would consist of the Data sets for hand gesture Detection and recognition. Here in, involves the data preparations which will be done through pandas for better computations and also preparations of data frames have to take place.

Training Model:

The data preparation is followed by the training of the given model. The model has to be trained for multiple numbers of gestures from the user to detect, understand and be able to interpret the user command. This might involve the use of Deep learning algorithm that is, CNN(Convolution Network (ConvNet/CNN) which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other, Pre-processing required in a ConvNet is much lower as compared to other classification algorithms.

Decision Making:

Based on the Gestures captured and recognized decision making plays an important role, the System has to make the correct decision as for the desired command to be executed which was given by the user.

Connectivity:

The target application for the project is PowerPoint presentation and the entire scope is limited to it, so whatever the decision is made by the system will make its way through for the connectivity to the targeted application which is PowerPoint presentation in our case.

Implementation:

Once the gestures are imposed on the application(ppt), its implementation takes place and further the application is controlled according to the gestures fed to the system.

CHAPTER 5

MODULES OF THE PROJECT

5.1 Modules included the project are as follows:

5.1.1. Downloading the Datasets and data preparation: It involves gathering of data , discover and access data ,cleanse, transform and then finally store data. Data preparation is the act of manipulating (or pre-processing) raw data (which may come from disparate data sources) into a form that can readily and accurately be analyzed, e.g. for business purposes. Data preparation is the first step in data analytics projects and can include many discrete tasks such as loading data or data ingestion, data fusion, data cleaning, data augmentation, and data delivery.

5.1.2. Training of Model: Here, system would be taught how to work .
It is divided into three sections:

1] Data Augmentation: -

Data augmentation in data analysis are techniques used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. It acts as a regularizer and helps reduce overfitting when training a machine learning model. It is closely related to oversampling in data analysis.

Data augmentation can be adequately used to prepare the Deep Learning models in such applications. A portion of the straightforward changes applied to the picture are; mathematical changes like to flip an image, rotate, transform or crop an image, scaling and shade spacing.

Figure Shows the original image and the images after applying some of the transformation:

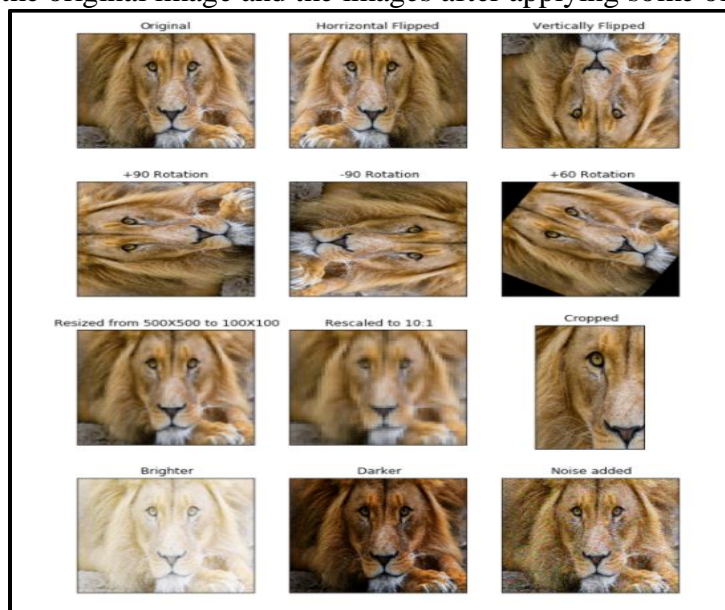


Fig 5.1: Some common image transformation applied for data augmentation

Some of the popular open-source python packages used for image augmentation packages are Albumentations, Keras ImageDataGenerator, OpenCV, Skimage.

The basic changes displayed in figure-can be accomplished utilizing any of these packages. Other than the straightforward ones, each package offers some custom changes.

Pictures could be shot in different conditions. The straightforward changes recorded above will be unable to represent every one of the varieties in the conditions.

2] Convolutional Neural Network (CNN) :-

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used.

CNN image classifications take an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension). Eg., An image of $6 \times 6 \times 3$ array of matrix of RGB (3 refers to RGB values) and an image of $4 \times 4 \times 1$ array of matrix of grayscale image.

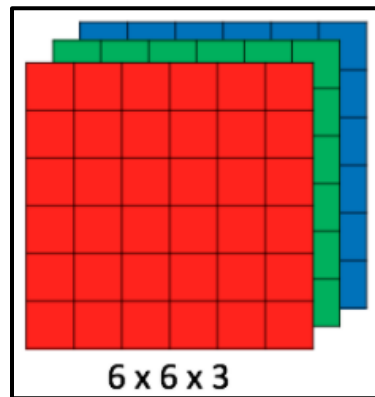


Fig: 5.2 Array of RGB Matrix

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

Convolution Layer

Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel.

Convolution of an image with different filters can perform operations such as edge detection, blur and sharpen by applying filters. The below example shows various convolution image after applying different types of filters (Kernels).








Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Fig 5.3: Some Common Filters Strides

Strides

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on. The below figure shows convolution would work with a stride of 2.

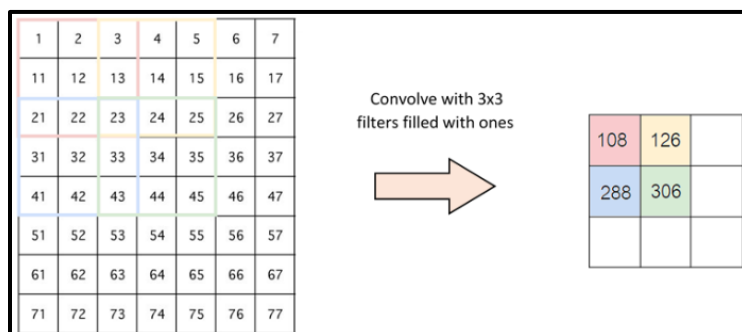


Fig 5.4 Stride of 2 pixels

Padding

Sometimes filter does not fit perfectly fit the input image. We have two options:

- Pad the picture with zeros (zero-padding) so that it fits
- Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.

Non-Linearity (ReLU):

ReLU stands for Rectified Linear Unit for a non-linear operation. The output is $f(x) = \max(0, x)$. Why ReLU is important: ReLU’s purpose is to introduce non-linearity in our ConvNet. Since, the real-world data would want our ConvNet to learn would be non-negative linear values.

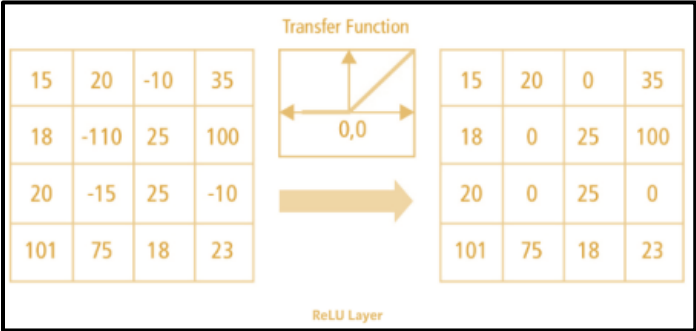


Fig 5.5: ReLU Operation

There are other nonlinear functions such as tanh or sigmoid that can also be used instead of ReLU. Most of the data scientists use ReLU since performance wise ReLU is better than the other two.

Pooling Layer:

Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling also called subsampling or down sampling which reduces the dimensionality of each map but retains important information. Spatial pooling can be of different types:

- Max Pooling
- Average Pooling
- Sum Pooling

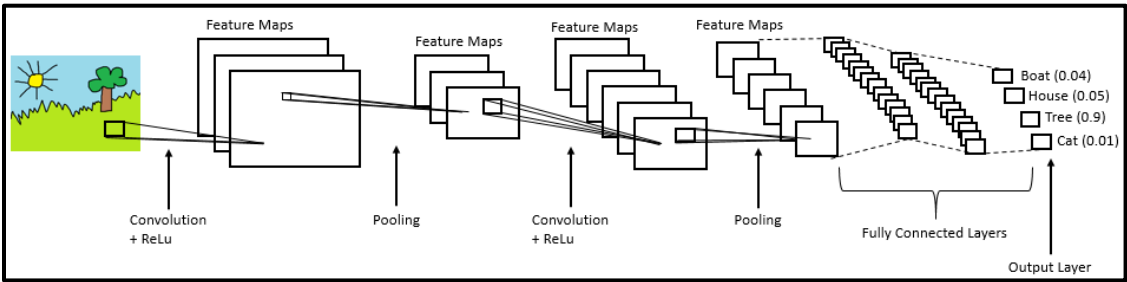


Fig 5.6 Complete CNN Architecture

Convolutional neural networks are comprised of two very simple elements, namely convolutional layers and pooling layers.

For these image processing problems there are numerous ways for arrangement of these layers.

Residual Network or ResNet:

The ultimate evaluation we will review on CNN was introduced by Kaiming He, et al. in their 2016 paper titled “Deep Residual Learning for Image Recognition.” In the paper, the authors proposed a very deep model called a Residual Network, or ResNet for short, an example of which achieved success on the 2015 version of the ILSVRC challenge

Their model had an impressive 152 layers. These are simply connections in the network architecture where the input is kept as-is (not weighted) and passed on to a deeper layer, e.g., skipping the next layer.

A residual block is a pattern of two convolutional layers with ReLU activation where the output of the block is combined with the input to the block, e.g., the shortcut connection.

The authors start with what they call a plain network, which is a VGG-inspired deep convolutional neural network with small filters (3×3), grouped convolutional layers followed with no pooling in between, and an average pooling at the end of the feature detector part of the model prior to the fully connected output layer with a softmax activation function.

Below picture analyze and contrast left to right the VGG architecture.

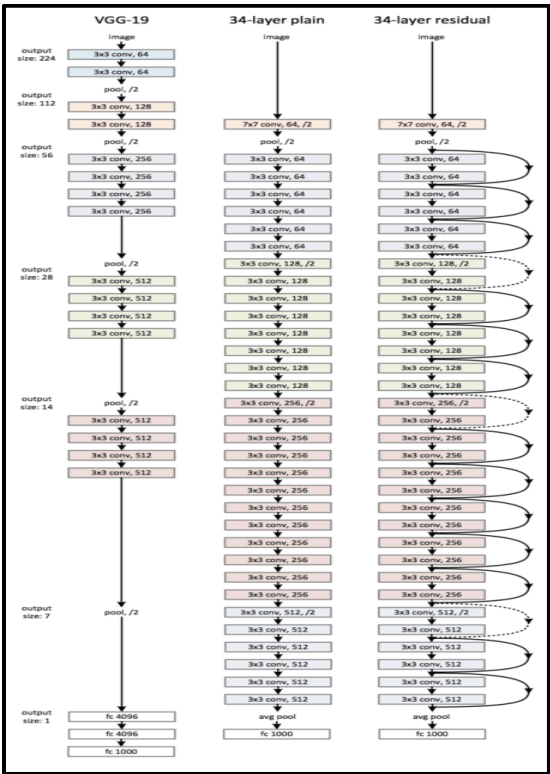


Fig 5.7 Architecture of Residual Network for Object photo Classification

Layer Function

We created this function to provide connectivity to the training model i.e. We piped some functions from our data augmentation file and our CNN and ResNet model file.

The functions which were piped are as given below to initialize the training file of our system

- Resnet3DBuilder function – to create our Resnet 101 model for the system
- Build_Resnet_101 function – to build Resnet model for the system.

Before proceeding, we shall explain SGD optimizer

- SGD (Stochastic gradient descent)
Stochastic gradient descent is an optimization algorithm often used in machine learning applications to find the model parameters that correspond to the best fit between predicted

and actual outputs. It's an inexact but powerful technique.

- Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

The Training is done with the help of given code below:

```
In [ ]: from Lib.data_loader import DataLoader
        from Lib.resnet_model import Resnet3DBuilder
        from Lib.HistoryGraph import HistoryGraph
        import Lib.image as img
        from Lib.Utils import makedirs
        import os

In [ ]: from math import ceil

In [ ]: from keras.optimizers import SGD

In [ ]: from keras.callbacks import ModelCheckpoint

In [ ]: target_size = (64,96)
        nb_frames = 16
        skip = 1
        nb_classes = 27
        batch_size = 64
        input_shape = (nb_frames,) + target_size + (3,)

In [ ]: workers = 8
        use_multiprocessing = False
        max_queue_size = 20

In [ ]: data_root = r'D:\Gesture Control System Dataset'
        csv_labels = r'D:\Gesture Control System Dataset\jester-v1-labels.csv'
        csv_train = r'D:\Gesture Control System Dataset\jester-v1-train.csv'
        csv_vale = r'D:\Gesture Control System Dataset\jester-v1-validation.csv'
        csv_test = r'D:\Gesture Control System Dataset\jester-v1-test.csv'
        data_vid = r'D:\Gesture Control System Dataset\Videos'
        model_name = 'resnet_3d_model'
        data_model = r'D:\Gesture Control System Dataset\model'

In [ ]: path_model = os.path.join(data_root, data_model, model_name)
        data_vid = os.path.join(data_root, data_vid)
        path_labels = os.path.join(data_root, csv_labels)
        path_train = os.path.join(data_root, csv_train)
        path_vale = os.path.join(data_root, csv_vale)
        path_test = os.path.join(data_root, csv_test)

In [ ]: data = DataLoader(data_vid, path_labels, path_train, path_vale)
        makedirs(path_model, 0o755)
        makedirs(os.path.join(path_model, 'graphs'), 0o755)

In [ ]: gen = img.ImageDataGenerator()
        gen_train = gen.flow_video_from_dataframe(data.train_df, data_vid, path_classes=path_labels, x_col='video_id', y_col='labels', t
        gen_vale = gen.flow_video_from_dataframe(data.vale_df, data_vid, path_classes=path_labels, x_col='video_id', y_col='labels', tar
```

```
In [ ]: resnet_model = Resnet3DBuilder.build_resnet_101(input_shape, nb_classes, drop_rate = 0.5)
        optimizer = SGD(lr=0.01, momentum=0.9, decay=0.0001, nesterov=False)
        resnet_model.compile(optimizer = optimizer, loss="categorical_crossentropy", metrics=["accuracy"])
        model_file = os.path.join(path_model, 'resnetmodel.hdf5')

In [ ]: model_checkpointer = ModelCheckpoint(model_file, monitor='val_acc', verbose=1, save_best_only=True, mode='max')

In [ ]: history_graph = HistoryGraph(model_path_name = os.path.join(path_model, "graphs"))

In [ ]: nb_sample_train = data.train_df["video_id"].size
        nb_sample_val = data.vale_df["video_id"].size

In [ ]: resnet_model.fit_generator(
        generator = gen_train,
        steps_per_epoch = ceil(nb_sample_train/batch_size),
        epochs = 100,
        validation_data = gen_vale,
        validation_steps = 30,
        shuffle = True,
        verbose = 1,
        workers = workers,
        max_queue_size = max_queue_size,
        use_multiprocessing = use_multiprocessing,
        callbacks = [model_checkpointer, history_graph])
```

After training our file by passing 16 frames and 32 frames. For 16 frames, we train our file for 50 epochs and for 32 frames we train our file for 100 epochs which after 46 epochs was not showing any improvement so we stopped our model for 32 frames at 46th epoch.

At this point, our accuracy was good about 99% and loss was also minimized when we pass 16 frames in our training model

```
Epoch 49/50
1418/1418 [=====] - 7270s 5s/step - loss: 4.6908 - acc:
0.9986 - val_loss: 5.0395 - val_acc: 0.9371

Epoch 0049: val_acc did not improve from 0.93783
Epoch 50/50
1418/1418 [=====] - ETA: 42:46 - loss: 4.6739 - acc: 0.9988
1418/1418 [=====] - 7375s 5s/step - loss: 4.6303 - acc:
0.9987 - val_loss: 5.0394 - val_acc: 0.9371
```

Fig 5.8: Epochs while passing 16 frames of the dataset

When we pass 32 frames in our training model, after about 46 epochs it was not improving and accuracy was also not improving so it stopped at 46th epoch giving accuracy about 99.42% of our training model.

```
Epoch 0044: val_accuracy did not improve from 0.89898
Epoch 45/100
2302/2302 [=====] - 1234s 536ms/step - loss: 4.1083 -
accuracy: 0.9950 - val_loss: 4.3859 - val_accuracy: 0.8922

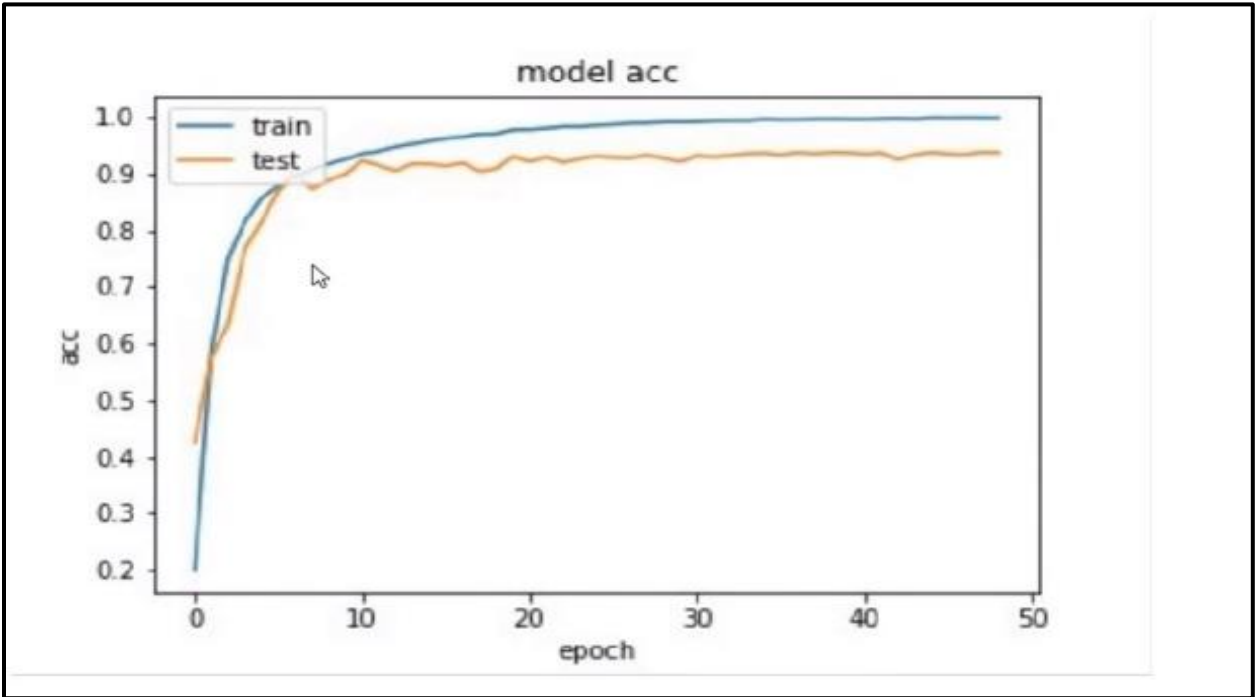
Epoch 0045: val_accuracy did not improve from 0.89898
Epoch 46/100
2302/2302 [=====] - 1234s 536ms/step - loss: 4.0766 -
accuracy: 0.9942 - val_loss: 5.0124 - val_accuracy: 0.8800

Epoch 0046: val_accuracy did not improve from 0.89898
```

Fig 5.9: Epochs while passing 32 frames of the dataset

The given below is the graphical representation of the above statistics for accuracy and loss of our training model:

Accuracy Graph:



Loss Graph:

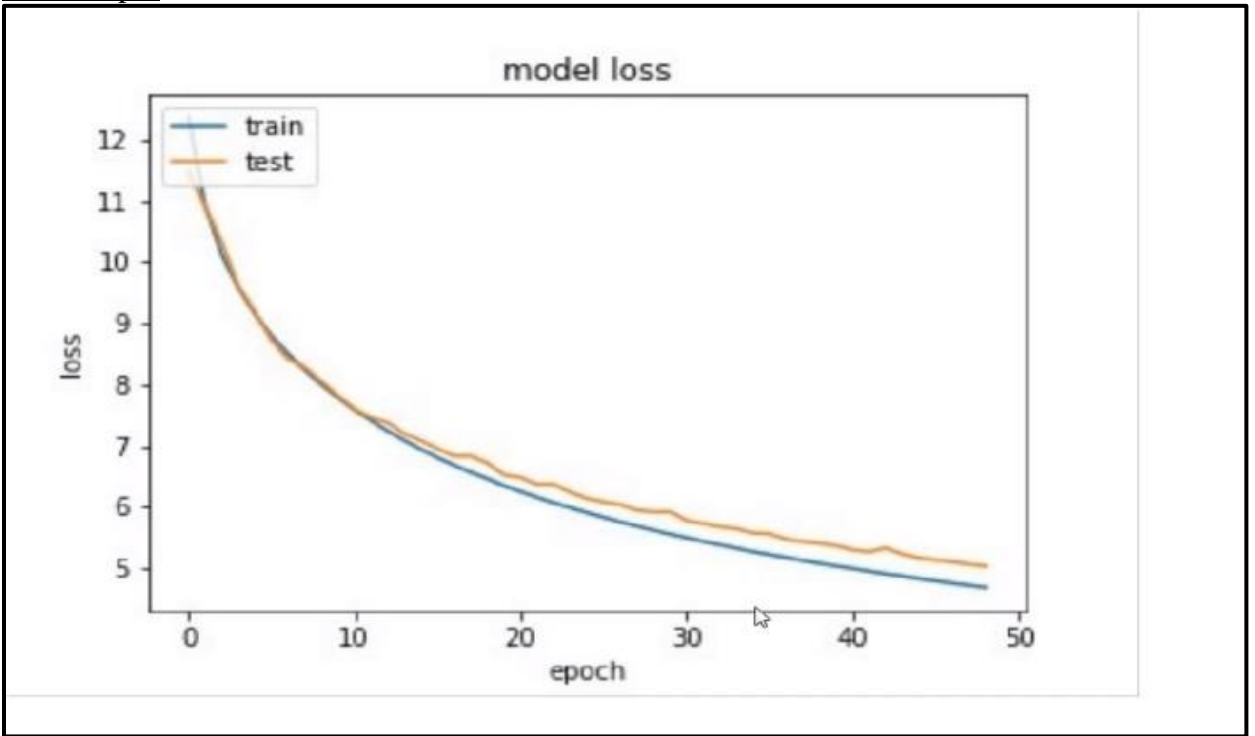


Fig 5.10: Accuracy and Loss graph

5.1.3. Classification: Classification refers to predictive modeling problems that involve predicting a class label or probability of class labels for a given input. Involves categorizing things based on properties.

After training the model, we need to classify the training sets to particular functions in our case we have to classify all the 27 gestures from the trained model.

After classifying all the gestures, we provide connectivity to the indented application i.e. we provide the trained and classified gestures to the indented application (in our project the media player) for automation purposes.

The classification is done with the help of given code below:

```
In [ ]: import cv2

In [ ]: from keras.models import load_model
import keras
import numpy as np
import pandas as pd

In [ ]: model = load_model(r"D:\Gesture Control System Dataset\model\resnetmodel.hdf5")

In [ ]: vid = cv2.VideoCapture(0)
vid.set(cv2.CAP_PROP_FRAME_WIDTH, 400)
vid.set(cv2.CAP_PROP_FRAME_HEIGHT, 400)

In [ ]: labels = pd.read_csv(r"D:\Gesture Control System Dataset\jester-v1-labels.csv", header = None)
```

```
In [ ]: buffer = []
cls = []
predicted_value = 0
final_label = ""
i = 1
while(vid.isOpened()):
    ret, frame = vid.read()
    if ret:
        image = cv2.resize(frame, (96,64))
        image = image/255.0
        buffer.append(image)
        if(i%16 == 0):
            buffer = np.expand_dims(buffer,0)
            predicted_value = np.argmax(model.predict(buffer))
            cls = labels.iloc[predicted_value]
            print(cls)
            print(cls.iloc[0])
            if(predicted_value == 0):
                final_label = "Swiping Left"
            elif(predicted_value == 1):
                final_label = "Swiping Right"
            elif(predicted_value == 2):
                final_label = "Swiping Down"
            elif(predicted_value == 3):
                final_label = "Swiping Up"
            elif(predicted_value == 4):
                final_label = "Pulling Hand Away"
            elif(predicted_value == 5):
                final_label = "Pulling Hand In"
            elif(predicted_value == 6):
                final_label = "Sliding Two Fingers Left"
            elif(predicted_value == 7):
                final_label = "Sliding Two Fingers Right"
```

```

elif(predicted_value == 8):
    final_label = "Sliding Two Fingers Down"
elif(predicted_value == 9):
    final_label = "Sliding Two Fingers Up"
elif(predicted_value == 10):
    final_label = "Pulling Two Fingers Away"
elif(predicted_value == 11):
    final_label = "Pulling Two Fingers In"
elif(predicted_value == 12):
    final_label = "Rolling Hand Forward"
elif(predicted_value == 13):
    final_label = "Rolling Hand backward"
elif(predicted_value == 14):
    final_label = "Turning Hand Clockwise"
elif(predicted_value == 15):
    final_label = "Turning Hand Counterclockwise"
elif(predicted_value == 16):
    final_label = "Zooming In With Full Hand"
elif(predicted_value == 17):
    final_label = "Zooming Out With Full Hand"
elif(predicted_value == 18):
    final_label = "Zooming In With Two Fingers"
elif(predicted_value == 19):
    final_label = "Zooming Out With Two Fingers"
elif(predicted_value == 20):
    final_label = "Thumb Up"
elif(predicted_value == 21):
    final_label = "Thumb Down"
elif(predicted_value == 22):
    final_label = "Shaking Hand"
elif(predicted_value == 23):
    final_label = "Stop Sign"
elif(predicted_value == 24):
    final_label = "Drumming Fingers"
elif(predicted_value == 25):
    final_label = "No gesture"
else:
    final_label = "Doing other things"
cv2.imshow('frame', frame)
buffer = []
i = i+1
text = "Activity: {}".format(final_label)
cv2.putText(frame, text, (20, 35), cv2.FONT_HERSHEY_SIMPLEX, 1.25, (0, 255, 0), 5)
cv2.imshow("frame", frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
vid.release()
cv2.destroyAllWindows()

```

For this we provide keyboard file i.e., keyboard press events from where you can call the functions from the keyboard.

This Keyboard file is used to give triggers to different keyboard commands without using the hardware keyboard.

After the setting up of the keyboard file we provide 7 trigger commands for the 9 gestures to control our media player.

This are the trigger codes for the given gestures:

```

VK_VOLUME_MUTE = 0xAD
VK_VOLUME_DOWN = 0xAE
VK_VOLUME_UP = 0xAF
VK_MEDIA_NEXT_TRACK = 0xB0
VK_MEDIA_PREV_TRACK = 0xB1
VK_MEDIA_PLAY_PAUSE = 0xB3
VK_MEDIA_STOP = 0xB2

```

These triggers codes will be connected to the gestures to control the windows media player functions
The automation is done with the help of the given code below:

```
jupyter Gesture Control Application Last Checkpoint: Last Saturday at 1:13 PM (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help Trusted | Gesture_Control_System O

In [ ]: import cv2

In [ ]: from keras.models import load_model
import keras
import numpy as np
import pandas as pd

In [ ]: model = load_model(r"D:\Gesture Control System Dataset\model\resnetmodel.hdf5")

In [ ]: vid = cv2.VideoCapture(0)
vid.set(cv2.CAP_PROP_FRAME_WIDTH, 400)
vid.set(cv2.CAP_PROP_FRAME_HEIGHT, 400)

In [ ]: labels = pd.read_csv(r"D:\Gesture Control System Dataset\jester-v1-labels.csv", header = None)

In [ ]: from keyboard import Keyboard

In [ ]: buffer = []
cls = []
predicted_value = 0
final_label = ""
i = 1
while(vid.isOpened()):
    ret, frame = vid.read()
    if ret:
        image = cv2.resize(frame, (96,64))
        image = image/255.0
        buffer.append(image)
        if(i%16 == 0):
            buffer = np.expand_dims(buffer,0)
            predicted_value = np.argmax(model.predict(buffer))
            cls = labels.iloc[predicted_value]
            print(cls)
            print(cls.iloc[0])
            if(predicted_value == 0):
                final_label = "Swiping Left"
                Keyboard.key(Keyboard.VK_MEDIA_NEXT_TRACK)
            elif(predicted_value == 1):
                final_label = "Swiping Right"
                Keyboard.key(Keyboard.VK_MEDIA_PREV_TRACK)
            elif(predicted_value == 2):
                final_label = "Swiping Down"
                Keyboard.key(Keyboard.VK_VOLUME_DOWN)
            elif(predicted_value == 3):
                final_label = "Swiping Up"
                Keyboard.key(Keyboard.VK_VOLUME_UP)
            elif(predicted_value == 8):
                final_label = "Sliding Two Fingers Down"
                Keyboard.key(Keyboard.VK_VOLUME_DOWN)
            elif(predicted_value == 9):
                final_label = "Sliding Two Fingers Up"
                Keyboard.key(Keyboard.VK_VOLUME_UP)
            elif(predicted_value == 20):
                final_label = "Thumb Up"
                Keyboard.key(Keyboard.VK_VOLUME_MUTE)
            elif(predicted_value == 23):
                final_label = "Stop Sign"
                Keyboard.key(Keyboard.VK_MEDIA_STOP)
            elif(predicted_value == 24):
                final_label = "Drumming Fingers"
                Keyboard.key(Keyboard.VK_MEDIA_PLAY_PAUSE)
            elif(predicted_value == 25):
                final_label = "No gesture"
            else:
                final_label = "Doing other things"
                cv2.imshow('frame', frame)
                buffer = []
            i = i+1
            text = "Activity: {}".format(final_label)
            cv2.putText(frame, text, (20, 35), cv2.FONT_HERSHEY_SIMPLEX, 1.25, (0, 255, 0), 5)
            cv2.imshow("frame", frame)
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
        vid.release()
        cv2.destroyAllWindows()
```

CHAPTER 6

REQUIREMENTS

6.1 HARDWARE REQUIREMENTS:

- RAM- 4GB
- Processor i5 and above
- Integrated web camera

6.2 SOFTWARE REQUIREMENTS:

6.2.1 Windows 8/10

6.2.2 Python v3.5

6.2.3 Anaconda

6.2.4 Jupyter Notebook

6.2.1 PYTHON LIBRARIES

- Keras package
- Pandas
- NumPy
- TensorFlow
- Matplotlib

CHAPTER 7

IMPLEMENTATION AND TESTING

7.1 IMPLEMENTATION:

After implementation of these three modules i.e.

- 1] Downloading the datasets and data preparation
- 2] Training of model
- 3] Classification

Now, we can go for our actual implementation of identifying of dynamic gestures and allocate the respective gesture specific command for automating the task for windows media player.

Following are some snapshots of our implementation work:

- 1] The Drumming Fingers Gesture is used to play the video in Windows Media Player.



Fig 7.1.1 – Gesture identification: Drumming Fingers

- 2] The Stop Sign is used to pause the video in Windows Media Player.

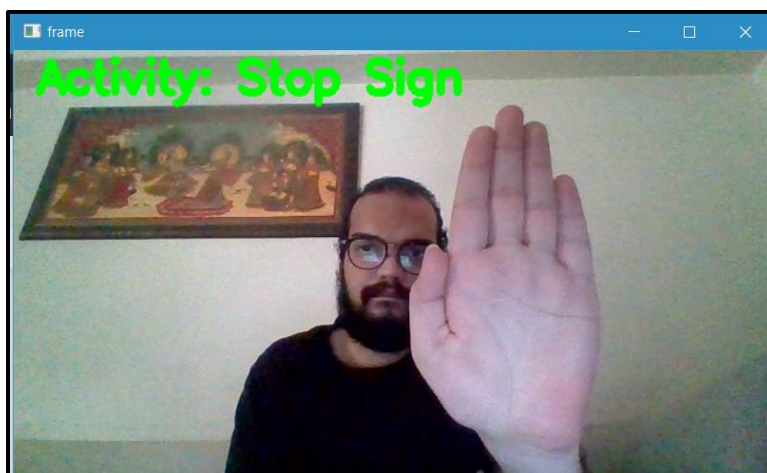


Fig 7.1.2 – Gesture identification: Stop Sign

3] The Swiping Up gesture is used to increase the volume of the Windows Media Player.



Fig 7.1.3 - Gesture identification: Swiping Up

4] The Swiping Down gesture is used to decrease the volume of the Windows Media Player.



Fig 7.1.4 - Gesture identification: Swiping Down

5] The Thumbs Up gesture is used to mute the volume of the Windows Media Player



Fig 7.1.5 - Gesture identification: Thumbs Up

6] The Swiping left gesture is used to skip the video forward in the Windows Media Player.



Fig 7.1.6 - Gesture identification: Swiping le

7] The Swiping Right gesture is used to skip the video reverse in the Windows Media Player.



Fig 7.1.7 - Gesture identification: Swiping Right

7.2 TESTING

A software testing technique also known as functional testing whereby the internal workings of the item being tested are not known by the tester. In a black box test on a software design, the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. The tester does not ever examine the programming code and does not need any further knowledge of the program other than its specifications.

The advantages of this type of testing include:

- The test is unbiased because the designer and the tester are independent of each other.
- The tester does not need knowledge of any specific programming languages.
- The test is done from the point of view of the user and not the designer.
- Test cases can be designed as soon as the specifications are complete

The disadvantages of testing include:

- The test can be redundant if the software designer has already run attest case.
- The test cases are difficult to design.
- Testing every possible input stream is unrealistic because it would take a inordinate amount of time; therefore, many program paths will go untested.

For a complete software examination, both White and black box tests are required. We used White box testing in our project.

CHAPTER 8

CONCLUSION AND

FUTURE SCOPE

CONCLUSION:

In proposed work, the gestured are used to control the Windows media player without the physical contact of hardware device like mouse. In previously implemented systems the costly 3D sensors were used, to cut down the investment cost of the system we this model uses the simple web camera.

The importance of gesture recognition lies in building efficient human- machine interaction. Thus, gesture recognition promises wide ranging applications in fields from photojournalism through medical technology to biometrics.

FUTURE SCOPE:

The model can be further expanded for other various applications like controlling power point presentations, gaming, sign language predictions and not only in software field but also in educational, medical and other sectors.

REFERENCES:

- [1] N. Krishna Chaitanya et R. Janardan Rao “Controlling OF Windows Media Player Using Hand Recognition System”, Journ. The International Journal Of Engineering And Science (IJES), vol. 3, PP 01- 04, 2014.
- [2] Neha S. Rokade, Harsha R. Jadhav, Sabiha A. Pathan, Uma Annamalai-“Controlling Multimedia Applications Using Hand Gesture Recognition.” 2015
- [3] M. Turk, “Gesture recognition,” Handbook of Virtual Environment Technology, 2001
- [4] OpenCV, “Mat,” <http://docs.opencv.org/modules/core/doc/core.html>”
- [5] TensorFlow, “<https://www.tensorflow.org>”
- [6] Keras, “<https://keras.io>”
- [7] Anaconda, “<https://www.anaconda.com>”
- [8] Jupyter Notebook, “<https://jupyter.org>”
- [9] Bruno Fernandes, Joaquin Fernández “Using Haar-like Feature Classifiers for Hand Tracking in Tabletop Augmented Reality”.
- [10] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in Computer Vision and Pattern Recognition, 2001.
- [11] W. Chen, “Real-time palm tracking and hand gesture estimation based on fore-arm contour,” 2011.
- [12] Data Augmentation, <https://www.mygreatlearning.com/blog/understanding-data-augmentation>
- [13] Convolutional Neural Network, https://en.wikipedia.org/wiki/Convolutional_neural_network
- [14] Residual Neural Network, https://en.wikipedia.org/wiki/Residual_neural_network

PROJECT GUIDE AND MEMBER'S INFORMATION

SR NO.	NAME	CONTACT	Email-Id
1.	Prof. Pallavi Wankhede	9970425086	pwankhede@stvincentngp .edu.in
2.	Akanksha Raut	9766964332	akanksharaut62@gmail.com
3.	Nidhi Dontulwar	9284360520	Dontulwarn@gmail.com
4.	Himanshu Bhole	8956218305	himanshubhole.30@gmail .com
5.	Hrishikesh Dhole	9730884996	hrishidh@gmail.com