A Project Report on

HAND GESTURE RECOGNITION

Submitted to the Dept. of Information Technology, SNIST
in the partial fulfillment of the academic requirements for the award of

B.Tech (Information Technology)

by

B.R.ASHRITH (18311A12C7)
P.AAKANKSHA REDDY (18311A12F2)
P.SATWIK (18311A12F3)

under the guidance of

Dr. N. Divya
Assistant Professor

Department of Information Technology
School of Computer Science and Informatics
Sreenidhi Institute of Science and Technology
Yamnampet, Ghatkesar Mandal, R. R. Dist., Hyderabad – 501301

Affiliated to
Jawaharlal Nehru Technological University Hyderabad
Hyderabad – 500 085

2022

Department of Information Technology

School of Computer Science and Informatics

Sreenidhi Institute of Science and Technology

Certificate

This is to certify that the Project report on "**Hand Gesture Recognition**" is a bonafide work carried out by Ashrith.B.R(18311A12C7), AakankshaReddy.P(18311A12F2), Satwik.P(18311A12F3) Science and Technology, Hyderabad, affiliated to Jawaharlal Nehru Technological University, Hyderabad under our guidance and supervision. The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Internal Guide	**Project Coordinator**	**Head of the Department**
(Dr.N.Divya)	(Dr.K.Kranthi Kumar)	(Dr.Sunil Bhutada)

External Examiner

DECLARATION

We,Ashrith.B.R(18311A12C7), AakankshaReddy.P(18311A12F2), Satwik.P(18311A12F3) Students of **Sreenidhi Institute of Science and Technology, Yamnampet, Ghatkesar,** Studying IV Year 1st semester, **Information Technology** solemnly declare that the major project work titled **Hand Gesture Recognition** is submitted to **Sreenidhi Institute of Science and Technology** for partial fulfillment for the award of degree of bachelor of technology in **Information Technology**.

It is declared to the best of our knowledge that the work reported does not form part of any dissertation submitted to any other University or Institute for award of any degree.

Acknowledgements

We would like to express our immense gratitude and sincere thanks to **Dr.C.V.Tomy** for giving us the opportunity to do our project work in **Sreenidhi Institute of science and technology**, Hyderabad. We would also like to express our deep felt gratitude and appreciation to **Dr.N.Divya, Assistant Professor** for her guidance, valuable suggestions and encouragement in completing the project work within the stipulated time.

We would like to express our sincere thanks **Dr. Ch. Shiva Reddy, Principal, Prof. Dr. Sunil Bhutada,** Head of the Department of Information Technology, **Dr.K.Kranthi Kumar**, Project work coordinator, Sreenidhi Institute of Science and Technology, Hyderabad for permitting us to do our project work at **Sreenidhi Institute of science and technology**, Hyderabad.

Finally, we would also like to thank the people who have directly or indirectly helped us and parents and friends for their cooperation in completing the project work.

Ashrith.B.R (18311A12C7)
AakankshaReddy.P (18311A12F2)
Satwik.P (18311A12F3)

ABSTRACT

Human Computer Interaction can acquire several advantages with the introduction of different natural forms of device free communication. Gestures are a natural form of actions which we often use in our daily life for interaction, therefore to use it as a communication medium with computers generates a new paradigm of interaction with computers. This paper implements computer vision and gesture recognition techniques and develops a vision based low cost input device for controlling the VLC player through gestures. VLC application consists of a central computational module which uses the Principal Component Analysis for gesture images and finds the feature vectors of the gestures.

The theoretical analysis of the approach shows how to do recognition in static background. The Training Images are made by cropping the hand gesture from static background by detecting the hand motion using Lucas Kanade Pyramidical Optical Flow algorithm. This hand gesture recognition technique will not only replace the use of mouse to control the VLC player but also provide different gesture vocabulary which will be useful in controlling the application.

In this we have discussed a low cost system which uses dynamic hand gesture recognition technique to control the VLC media player. This application contains a central computation module which segments the foreground part of the frame using skin detection and approximate median technique. This hand gesture recognition technique introduces a new, natural way to interact with computers.

CONTENTS

CHAPTERS **PAGE NO**

List of Figures

List of Abbrevations

Notation	Description
CNN	**Convolutional Neural Network**
PUI	**Perceptual User Interface**
HCI	**Human computer interaction**

1. INTRODUCTION

Computers and computerized devices have become an eminent element of our society. They increasingly influence many aspects of our lives; With the increase in interaction of computers in our daily life, it would be worthy enough to get a Perceptual User Interface (PUI) to interact with computers as human interact with each other. Vision-based gesture recognition is an important technology for friendly humancomputer interface, and has received more and more attention in recent years . The applications designed for gesture recognition generally require restricted background, set of gesture command and a camera for capturing images. The gesture used in application for performing an action must represent the action which is being performed by it and also it must be logically explainable, thus for controlling a media player like VLC dynamic hand gestures could be more intuitive and natural.

1.1 Overview

A VLC media player function that was controlled by hand gesture includes play, pause, Full screen, stop, increase volume, decrease volume. Lucas Kanade Pyramidical Optical Flow algorithm was used to detect hand from video. This algorithm detects moving points in the image

1.2 Objectives

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

1.3 Problem Formulation

When a user sometimes cannot change the volume, forward, backward,play and pause according to their mood. So, this project helps the user to do all the operations by hand gestureThis project helps to improve the music application by adding more realistic features. We us a deep learning CNN algorithm which is used to detect facial emotion. We capture images using OpenCV and CNN algorithm helps to detect the Hand gesture of the user.

1.4 Scope

The purpose of this document is to describe all external requirements for the E-learning System. It also describes the interfaces for the system.

1.5 Feasibility

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are :

Economical Feasibility
Technical Feasibility
Social Feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user.

This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

1.6 System Requirements

1.6.1 Software Requirements

- Programming Language : Python
- IDE : PyCharm, Jupyter
- UML Design : Start UML
- Tools : PIP

1.6.2 Hardware Requirements

- Processor : Intel core i3 and above
- RAM : 4GB and Higher
- Hard Disk : 500GB (Minimum)

2. REQUIREMENT SPECIFICATION

2.1.Functional Requirements

Here, the focus is on specifying what has been found giving analysis such as representation, Specification languages and tools, and checking the specifications are addressed during this activity. The requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic of this phase.

Role of SRS:

The purpose of the SRS is to reduce the communication gap between the clients and the developers. SRS is the medium though which the client and user needs are accurately specified. It forms the basis of software development. A good SRS should satisfy all the partiesinvolved in the system.

2.2 Constraints/Perceptons

- Coding using OpenCV and python
- uOpenCV 3.2.0 should be installed
- uPython v3.5 should be installed
- uMatplotlib 2.0 should be installed.
- uImporting required libraries from OpenCV
- uimutils==0.4.5
- unumpy==1.14.0
- uopencv-contrib-python=3.4.0.12
- uopencv-python==3.4.0.12
- uvlc-ctrl==1.0

3. ANALYSIS

3.1 Use Case diagram

In the Unified Modeling Language (UML), a use case diagram can summarize the details of our system's users (also known as actors) and their interactions with the system. To build one, we will use a set of specialized symbols and connectors.

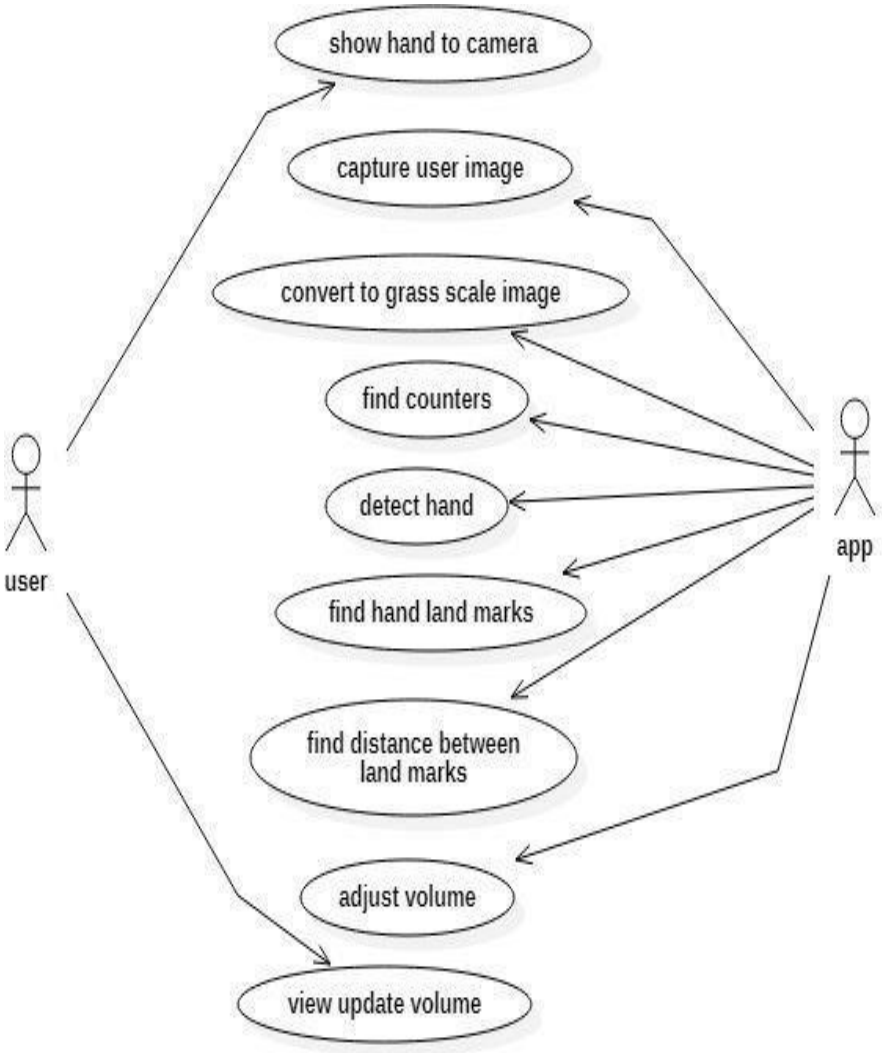


Figure 3.1: Use Case Diagram

3.2 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational stepby-step workflows of components in a system. An activity diagram shows the overall flow of control.

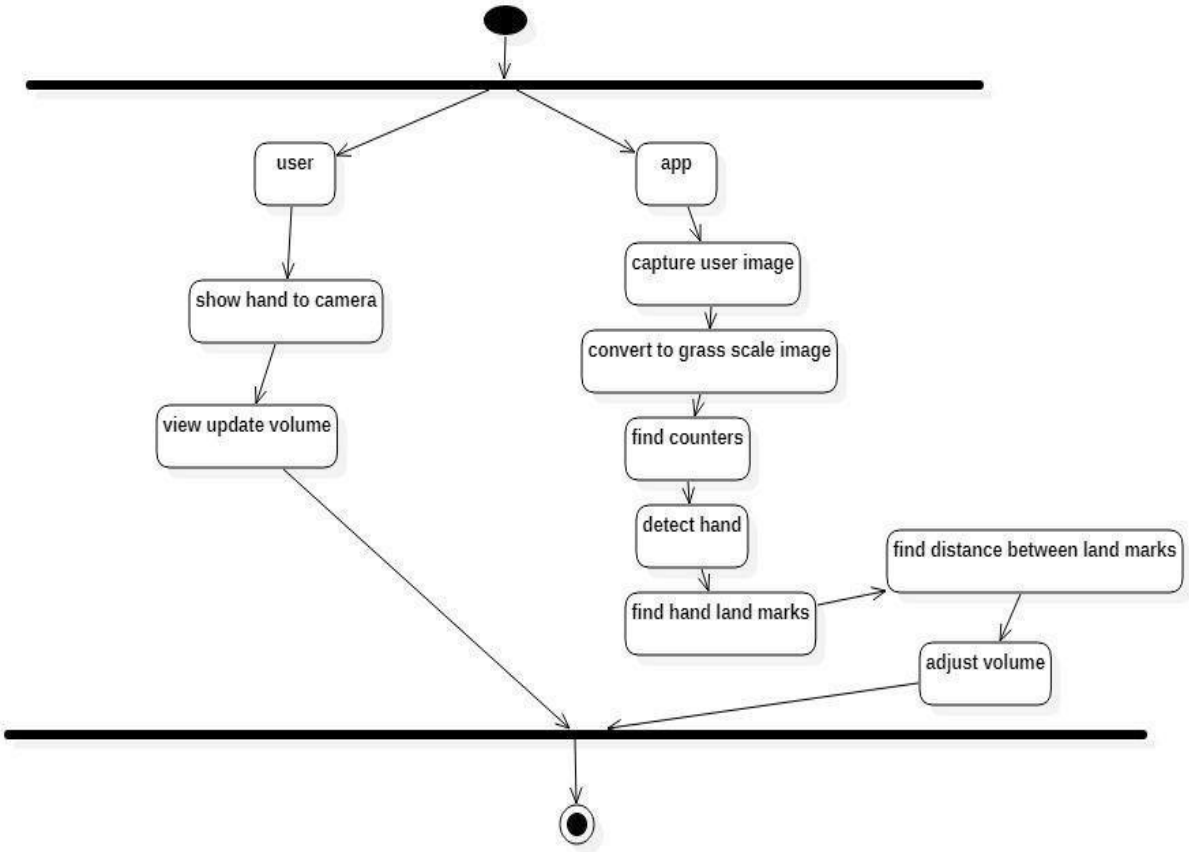
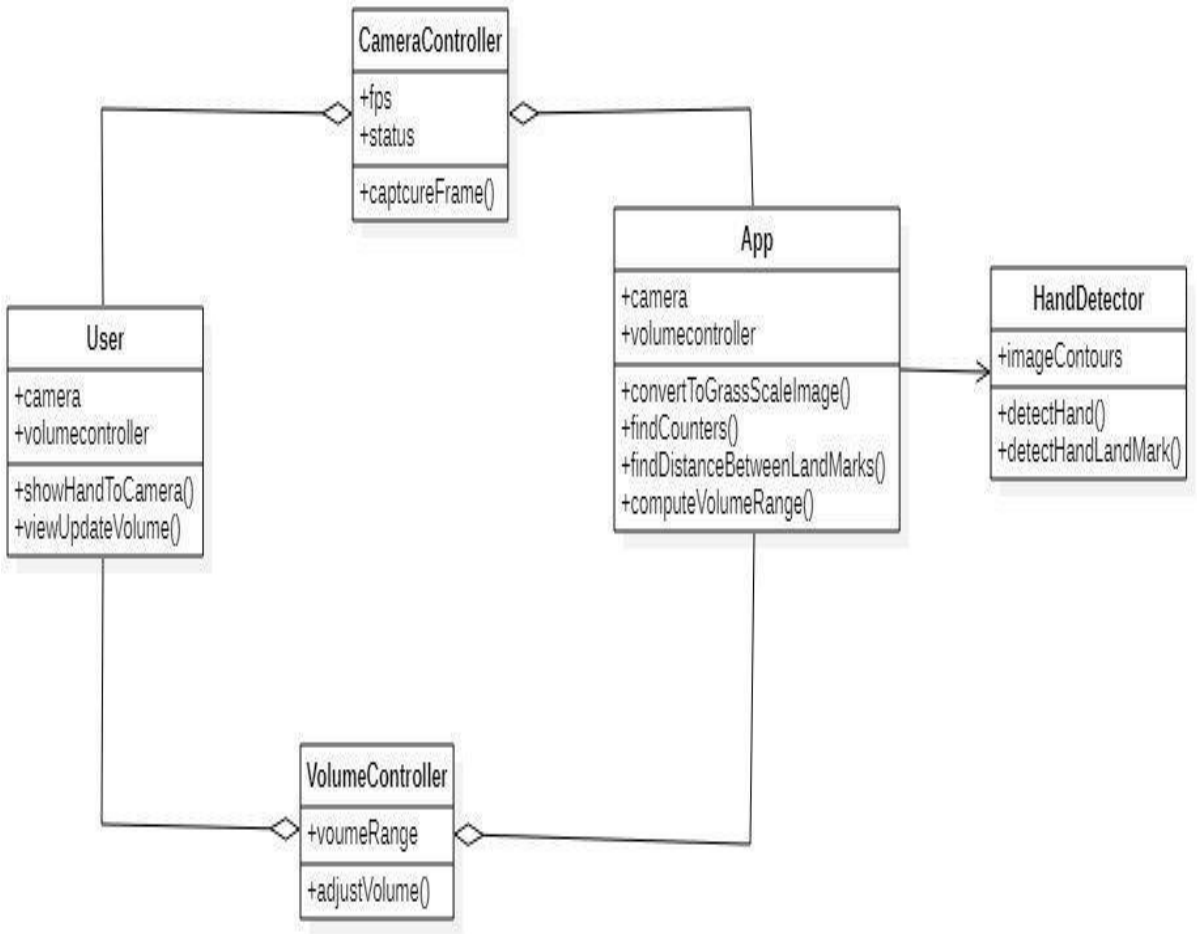


Figure 3.2: Activity Diagram

4.DESIGN

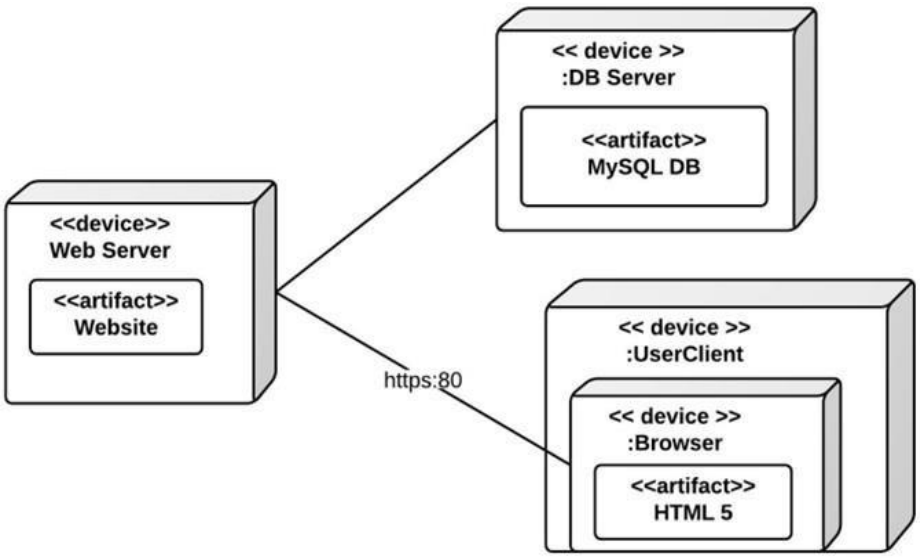
4.1 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



4.2 Deployment diagram

Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed.Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.



5. MODULES

Data acquisition: Data acquisition testing of our system was done by in the laptop.

Segmentation: Image segmentation techniques. The complete process w the next section.

Skin Detection Model for d region.

Approximate Median subtraction of background. It has been observed that by the methods for segmentation the blob better for further process.

Feature extraction: In our work we h direction of the hand region as a feat **Recognition**

Phase: In this work D used as a classification tool for gestures.

VLC interaction: Give the appropr the VLC player according to the recognization.

Pre-processing:

This is the first step performed in image processing. In this step the noise from the image is removed by using median filtering. Median filtering is one of the most widely used noise reduction technique. This is because in median filtering the edges in image are preserved while the noise is still removed.

1) Conversion to Gray-Scale: After the pre-processing step, the image is converted into grayscale. Conversion into grayscale is necessary because different writers use pens of different colours with varying intensities. Also working on grayscale images reduces the overall complexity of the system.

2) Thresholding: When an image is converted into grayscale, the handwritten text is darker as compared to its background. With the help of thresholding we can seperate the darker regions of the image from the lighter. Thus because of thres holding we can seperate the handwritten text from its background.

3) Image Segmentation: A user can write text in the form of lines. Thus the thresholded image is first segmented into individual lines. Then each individual line is segmented into individual words. Finally each word is segmented into individual characters. Segmentation of image into lines is carried out using Horizontal projection method. First the thresholded image is inverted so that background becomes foreground and vice-versa. Now the image is scanned from top to bottom. While scanning , the sum of pixels in each row of image is calculated. The sum of pixels will be zero if all the pixels in one particular row are black. The sum will be non-zero if some white pixels are present in a row. After this a horizontal histogram is plotted in

which the X-axis represents the Y-coordinate of image(Starting from Top to Bottom) and the Y-axis represents to the Y-coordinate.

6. IMPLEMENTATION

In this section we are going recognition process in the step by step manner

1. Capture 10 second video (V) data from the webcam.
2. Split the video into RGB frames.
3. For each RGB frame (Vi) Skin area was detected by using Skin Detection Model [6,8] and a new image was formed in the skin area marked by red color.Fig.3 shows an example of the image Visd produce by Skin Detection Model.
4. Approximate Median based segmentation [7,9] was done on the Vi and a binary image was obtained. Fig.4 shows an example of the image Viam produce by Approximate Median Techniques.
5. AND operation was performed on Visd and Viam for obtaining the final binary image for feature extraction. Fig.5 shows an example image Vif produces after the AND operation.

   Vif = Visd AND Viam
6. Various features were extracted from Vf array,like:

   Dox - direction of movement on X coordinate.

   Doy - direction of movement on Y coordinate.

   Area - Area was calculated which contained the hand of the user.

   Xstart - Starting position of the hand gesture in X coordinate.

   Xend - Ending position of the hand gesture in X coordinate.

   Ystart - Starting position of the hand gesture in Y coordinate.

   Yend - Ending position of the hand gesture in Y coordinate.
7. These features were used in a Decision Tree for recognition of the gestures in real time. The algorithm of the decision tree is given in the Fig.6 .

Libraries

Numpy: NumPy is a Python package which stands for ‘Numerical Python’. It is the core library for scientific computing, which contains a powerful n-dimensional array object, provide tools for integrating C, C++ etc. It is also useful in linear algebra, random number capability etc. NumPy array can also be used as an efficient multi-dimensional container for generic data. Now, let me tell you what exactly a python numpy array is.

To install Python NumPy, go to your command prompt and type “pip install numpy”. Once the installation is completed, go to your IDE (For example: PyCharm) and simply import it by typing: “import numpy as np”.

Pandas: Pandas are an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.
In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Standard Python distribution doesn't come bundled with Pandas module. A lightweight alternative is to install NumPy using popular Python package installer, pip. pip install pandas

Matplotlib: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits.

Seaborn: Seaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and closely integrated with pandas data structures.

Here is some of the functionality that seaborn offers:

- A dataset-oriented API for examining relationships between multiple variables
- Specialized support for using categorical variables to show observations or aggregate statistics
- Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data
- Automatic estimation and plotting of linear regression models for different kinds dependent variables
- Convenient views onto the overall structure of complex datasets
- High-level abstractions for structuring multi-plot grids that let you easily build complex visualizations.
- Concise control over matplotlib figure styling with several built-in themes
- Tools for choosing color palettes that faithfully reveal patterns in your data

Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

Sklearn: Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. This library, which is largely written in Python, is built upon NumPy, SciPy and Matplotlib.

Algorithm

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm 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. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

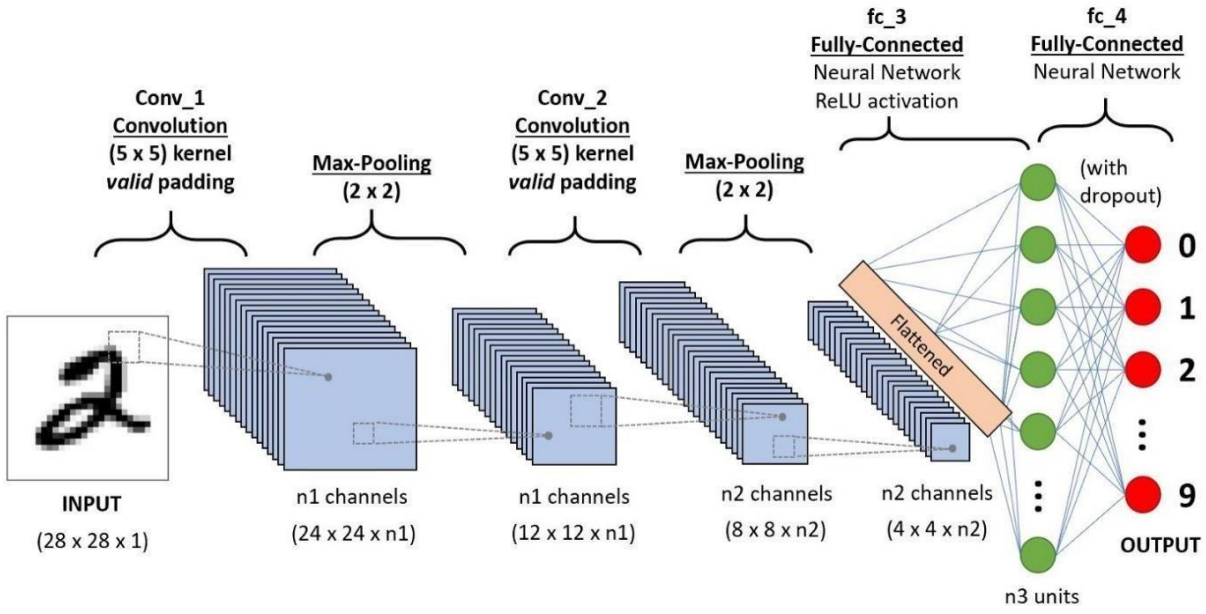


Fig 6.1

6.1. Source Code

```
import cv2
import mediapipe as mp
import time
import os
import pyautogui as p

class handDetector():
    def __init__(self, mode=False, maxHands=2, detectionCon=0.5, trackCon=0.5):
        self.mode = mode
        self.maxHands = maxHands
        self.detectionCon = detectionCon
        self.trackCon = trackCon

        self.mpHands = mp.solutions.hands
        self.hands = self.mpHands.Hands(self.mode, self.maxHands,
                                        self.detectionCon, self.trackCon)
        self.mpDraw = mp.solutions.drawing_utils

    def findHands(self, img, draw=True):
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.hands.process(imgRGB)
        # print(results.multi_hand_landmarks)

        if self.results.multi_hand_landmarks:
            for handLms in self.results.multi_hand_landmarks:
                if draw:
```

```
                self.mpDraw.draw_landmarks(img, handLms,
                                           self.mpHands.HAND_CONNECTIONS)
        return img

    def findPosition(self, img, handNo=0, draw=True):

        lmList = []
        if self.results.multi_hand_landmarks:
            myHand = self.results.multi_hand_landmarks[handNo]
            for id, lm in enumerate(myHand.landmark):
                # print(id, lm)
                h, w, c = img.shape
                cx, cy = int(lm.x * w), int(lm.y * h)
                # print(id, cx, cy)
                lmList.append([id, cx, cy])
                if draw:
                    cv2.circle(img, (cx, cy), 15, (255, 0, 255), cv2.FILLED)

        return lmList



wCam, hCam = 640, 480

cap = cv2.VideoCapture(0)
cap.set(3, wCam)
cap.set(4, hCam)

pTime = 0
```

```
detector =handDetector(detectionCon=0.75)

tipIds = [4, 8, 12, 16, 20]

while True:
    success, img = cap.read()
    img = detector.findHands(img)
    lmList = detector.findPosition(img, draw=False)
    print(lmList)

    if len(lmList) != 0:
        fingers = []

        # Thumb
        if lmList[tipIds[0]][1] > lmList[tipIds[0] - 1][1]:
            fingers.append(1)
        else:
            fingers.append(0)

        # 4 Fingers
        for id in range(1, 5):
            if lmList[tipIds[id]][2] < lmList[tipIds[id] - 2][2]:
                fingers.append(1)
            else:
                fingers.append(0)

# print(fingers)
```

```
    totalFingers = fingers.count(1)
    #print("Result:",totalFingers)

    result=""
    if totalFingers==1:
        p.press("space")
        result = "forward"
    elif totalFingers==2 :
        p.press("left")
        result="backward"
    elif totalFingers==3:
        p.press("right")
        result = "volume up"
    elif totalFingers==4:
        p.press("up")
        result = "volume down"
    elif totalFingers==5:
        p.press("down")
        result = "volume down"

    cv2.putText(img, str(result), (45, 375), cv2.FONT_HERSHEY_SIMPLEX,1, (255, 0, 0),3)

cTime = time.time()
fps = 1 / (cTime - pTime)
pTime = cTime

cv2.imshow("Image", img)
cv2.waitKey(1000)
```

7. SYSTEM ARCHITECTURE

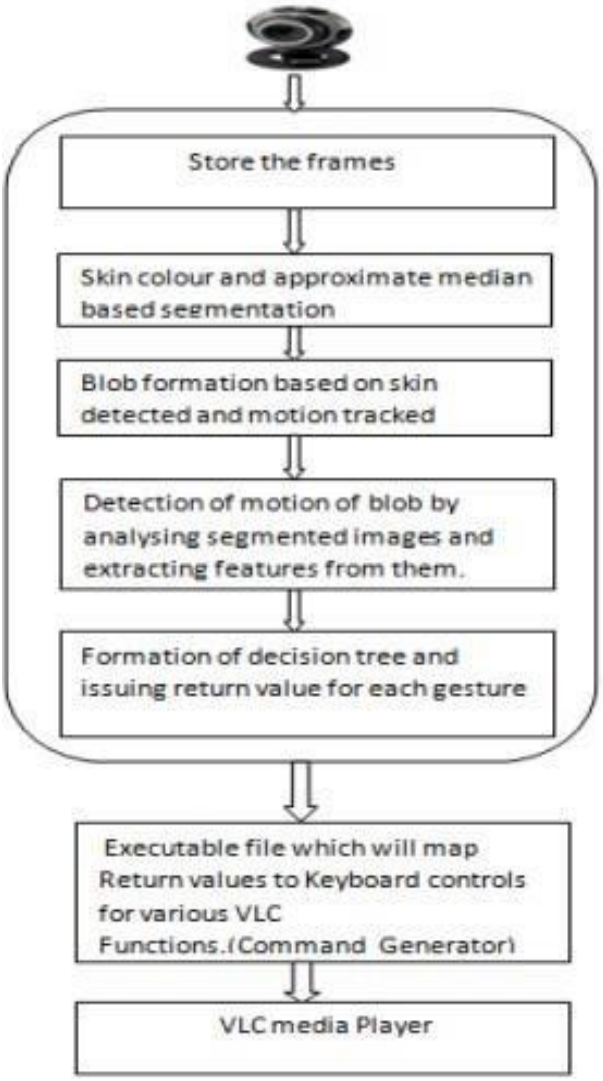


Fig 7.1 System Architecture

8. USER SCREEN

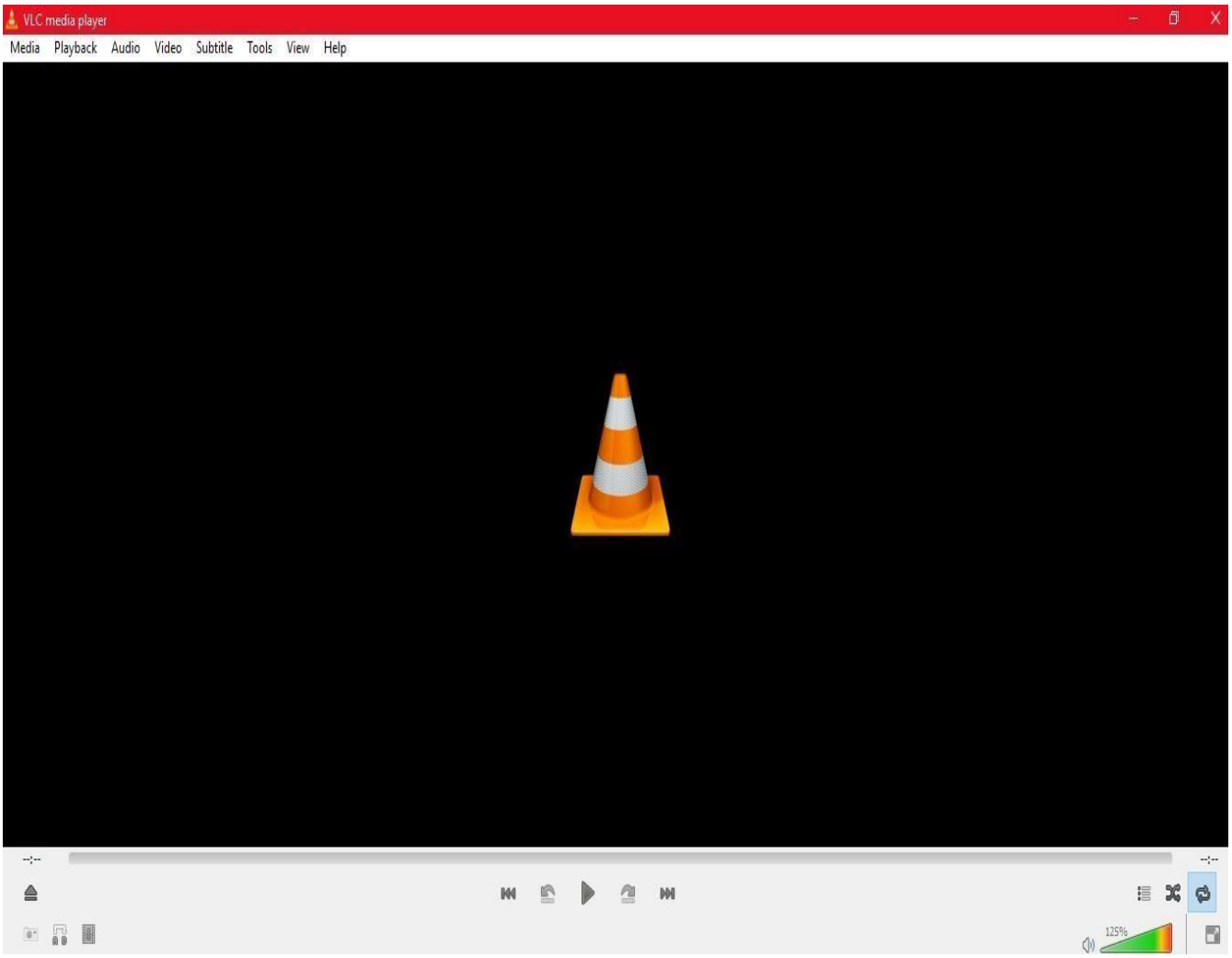


Fig 8.1 User screen

9. Testing Methodology

TYPES OF TESTS:

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

10. TESTING METHODOLOGY ON SAMPLE VIDEO

When we show the index finger as shown below it will make operation as pause /play.

Fig 10.1.1 video pause/play

When we show our two fingers it will foward for 5 seconds

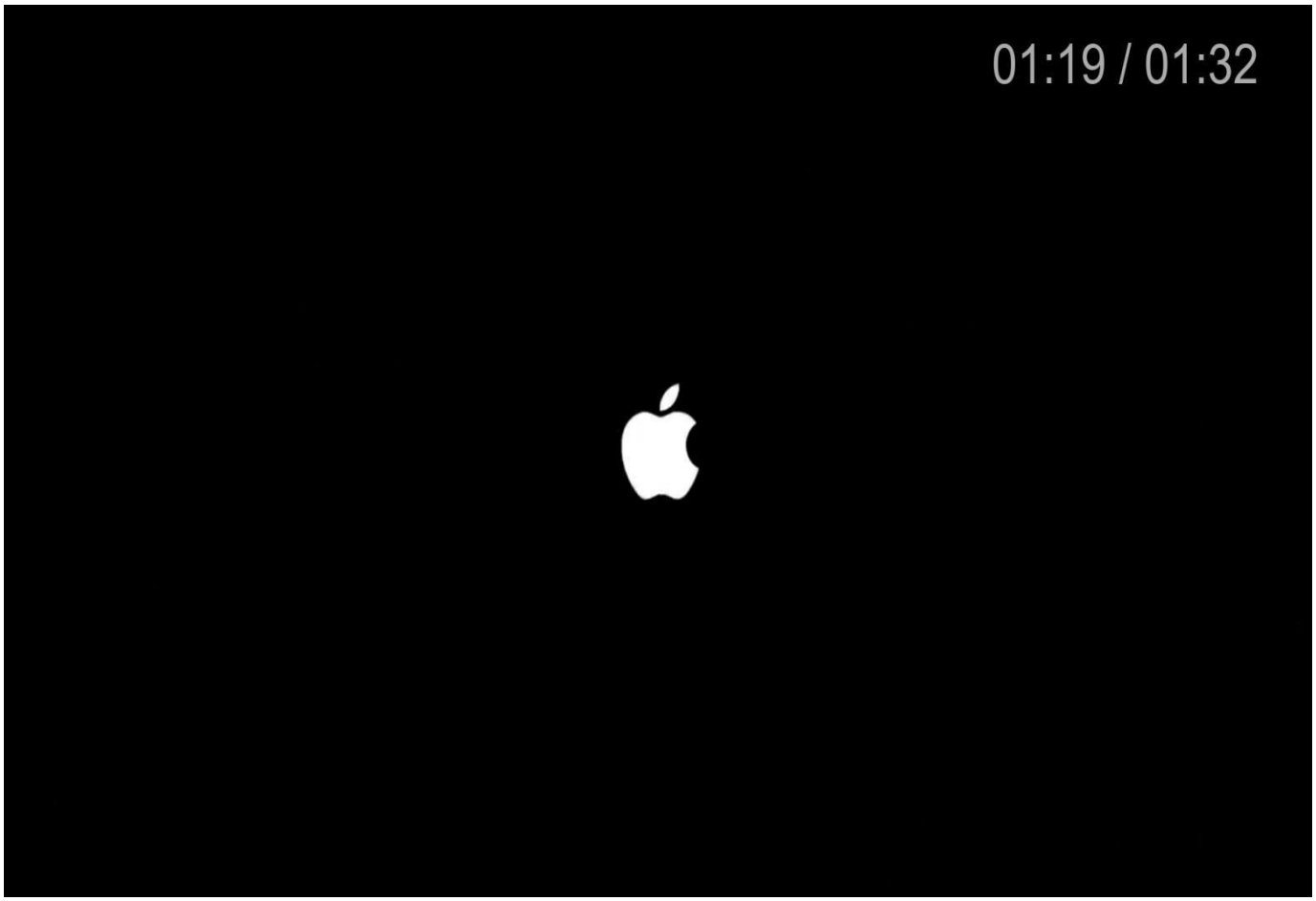


Fig 10.1.2
Video forwarded 5 sec

When we show three fingers it will move backwards for 5 seconds

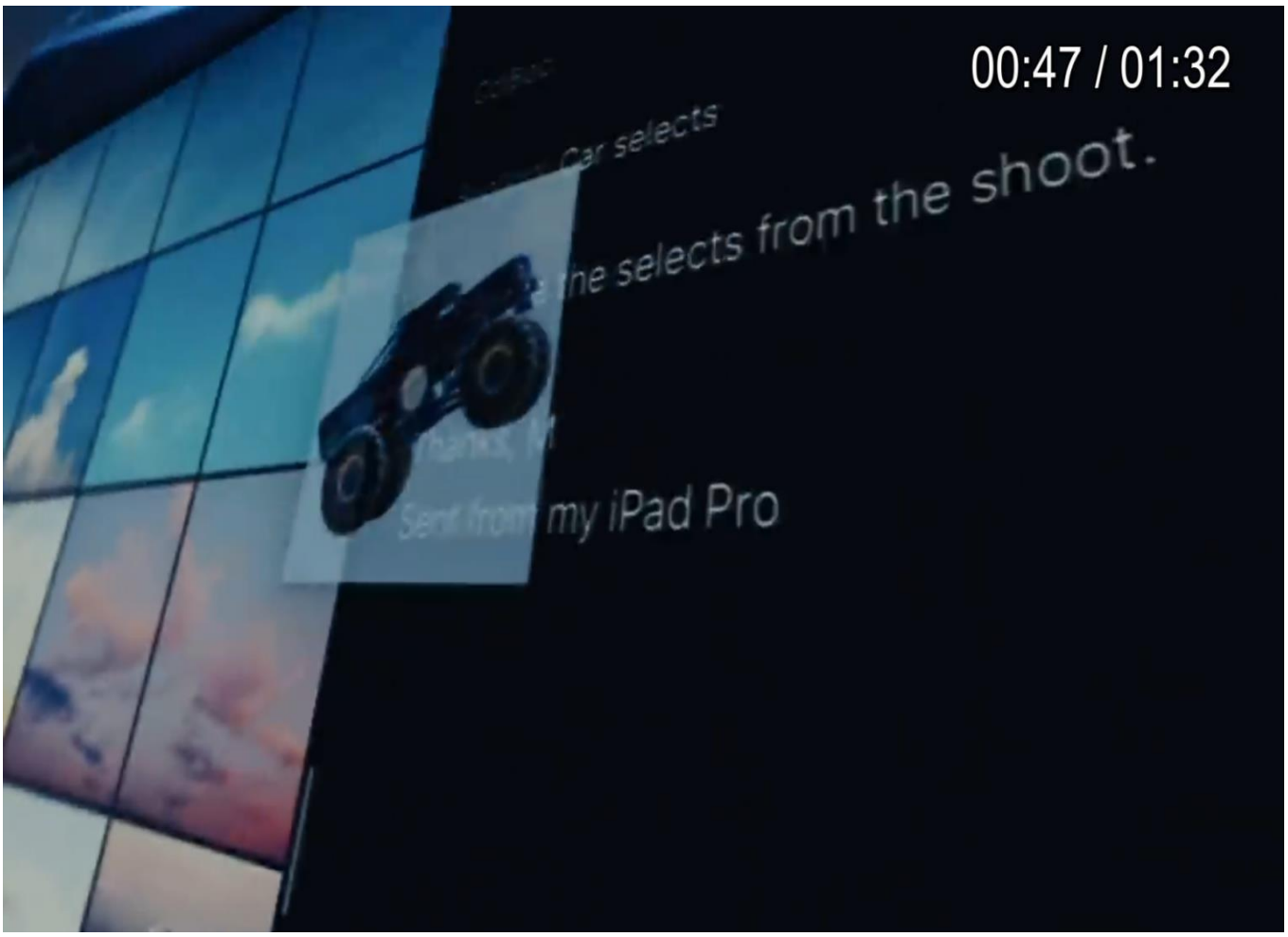


Fig 10.1.3
Video played backward
by 5 sec

When we show 4 fingers volume will be increased upto 5%

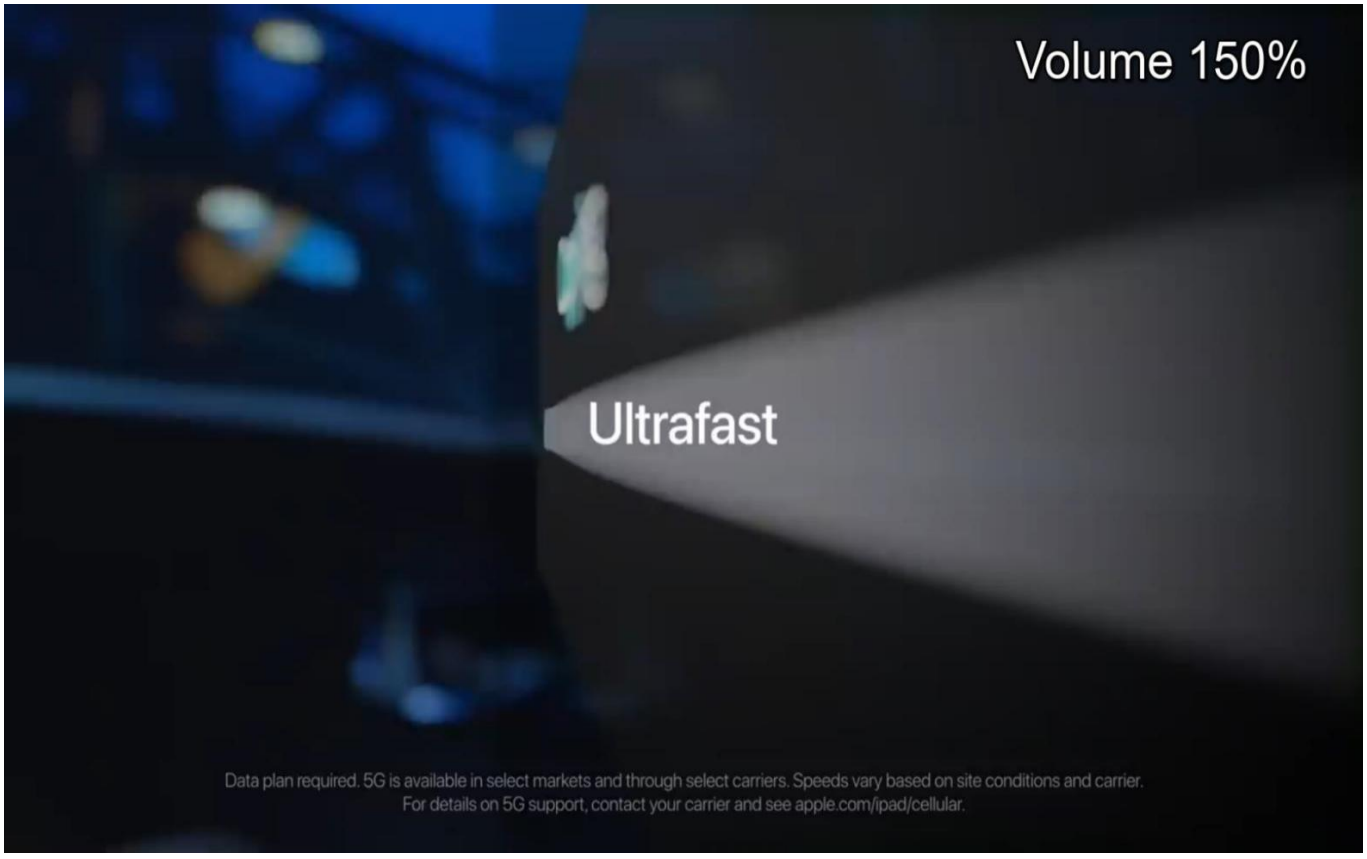


Fig10.1.4
Volume increased by 5%

When we show five fingers the volume will be decreased 5% per second.

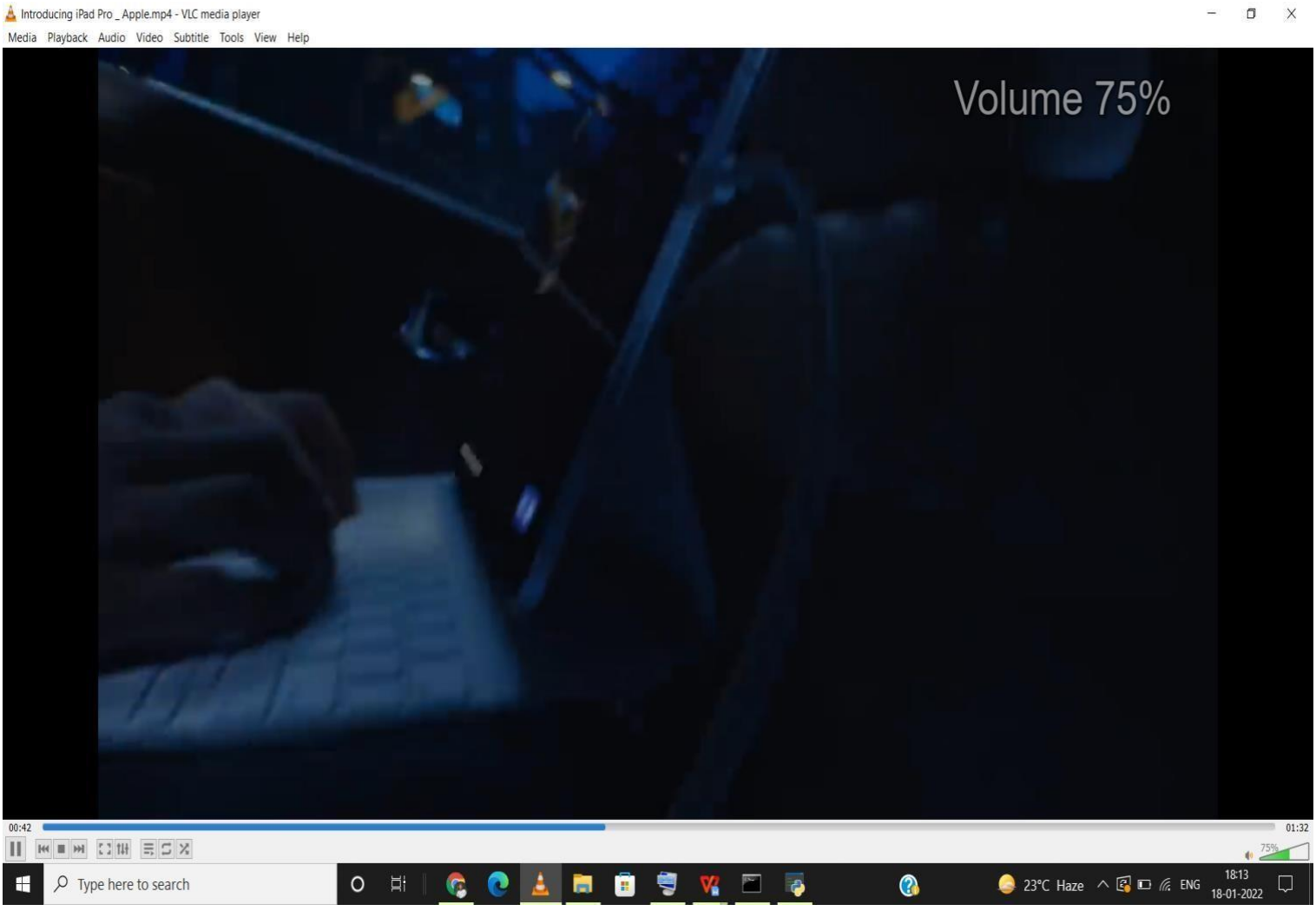


Fig 10.1.5

Volume decreased by 5%

11. RESULT

The system was tested in real time and very promising results. We have tested our for different gestures with different person some snap shots of our system. In Table I we recognition percentage of the different gesture in results comes because of the different no and different light illumination.

Table I: Recognition rate of different gestures.

Gesture	Recognition Rate (%)
Play / Pause	80%
Stop	80%
Volume Increase	70%

12. CONCLUSION

In current world many facilities are available for providing input to any application some needs physical touch and some without using physical touch (speech, hand gesture etc.). But not many applications are available which are controlled using current and smart facility of providing input which is by hand gesture .By this method user can handle application from distance without using keyboard and mouse. This application provides a novel human computer interface by which a user can control media player (VLC) using hand gesture. The application defines some gesture for controlling the functions of VLC player. The user will provide gesture as an input according to interested function. The application provides a flexibility of defining user interest gestures for specific command which make the application more useful for physically challenged people, as they can define the gesture according to their feasibility.

13.REFERENCES

[1] M. Turk and G. Robertson, “Perceptual user interfaces,” Communications of the ACM, vol. 43(3), March 2000.

[2] A. Nandy, S. Mondal, J. S. Prasad, P. Chakraborty and G. C. Nandi, “Recognizing & interpreting Indian Sign Language gesture for Human Robot Interaction,” 2010 International Conference on Computer and Communication Technology (ICCCT), pp. 712-717, IEEE, Sept. 2010.

[3] Y. Wu and T. S. Huang, "Vision-Based Gesture Recognition: A Review", Lecture Notes in Computer Science, Vol. 1739,pp. 103-115, 1999.

[4] V. Pavlovic, R. Sharma and T. S. Huang, “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review,” IEEE Trans. On Pattern Analysis and Machine Intelligence, vol. 19(7), pp. 677-695, 1997.

[5] S. S. Rautaray and A. Agrawal, “A Vision based Hand Gesture Interface for Controlling VLC Media Player,” International Journal of Computer Applications, vol. 10(7), pp. 11–16, November 2010.

[6] C. Ó. Conaire, N. E. O'Connor and A. F. Smeaton, “Detector adaptation by maximizing agreement between independent data sources,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1-6, June 2007.

[7] N. J. B. McFarlane and C. P. Schofield, “Segmentation and tracking of piglets in images,” Machine Vision and Applications, vol. 8, issue 3, pp. 187-193, 1995.

[8] M. J. Jones and J. M. Rehg, “Statistical color models with application to skin detection,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 274-280, 1999.

[9] S. Battiato, D. Cantone, D. Catalano and G. Cincotti; “An Efficient Algorithm for the

Approximate Median Selection Problem," Algorithms and Complexity Lecture Notes in Computer Science, Springer, vol. 1767, pp. 226-238, 2000.

[10] A. Nandy, J. S. Prasad, S. Mondal, P. Chakraborty and G. C. Nandi, "Recognition of Isolated Indian Sign Language Gesture in Real Time," Information Processing and Management, Communications in Computer and Information Science, Springer, vol. 70, pp. 102-107, 2010.

[11] A. Nandy, J. S. Prasad, P. Chakraborty, G. C. Nandi, S. Mondal, "Classification of Indian Sign Language In Real Time," International Journal on Computer Engineering and Information Technology (IJCEIT), vol. 10, issue 15, 2010.