Media Engineering and Technology Faculty
German University in Cairo

Hand Gesture Recognition

Bachelor Thesis

Author:	Farah Ahmed Hassan
Supervisors:	Assoc. Prof. Milad Michel Ghantous
Submission Date:	1 August, 2021

Media Engineering and Technology Faculty
German University in Cairo

Hand Gesture Recognition

Bachelor Thesis

Author:	Farah Ahmed Hassan
Supervisors:	Assoc. Prof. Milad Michel Ghantous
Submission Date:	1 August, 2021

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowlegement has been made in the text to all other material used

Farah Ahmed Hassan
1 August, 2021

Acknowledgments

First of all, I would like to thank Dr. Milad Micheal Ghantous, who assisted me throughout this process with his constructive feedback, comfortable workspace, patience, motivation, enthusiasm and knowledge. Secondly, Furthermore, I would like express my sincere gratitude to my parents, who always supported me under all circumstances, no matter how difficult they were.

Abstract

Music players are mostly used in vehicle which required interaction between human and music player system for access and control. It is reported that handling music player system in car while driving is one of the reason to distract drivers. This leads to severe accidents which are possible to life threatening.To overcome the issue, hand gesture is the key to control music player system in vehicles. We developed hand gesture based music player control system in car using camera. This system helps to access all functions of music player like skipping , playing , pausing and increasing and decreasing the volume of the songs .

Contents

Chapter 1

Introduction

1.1 Motivation

The main purpose of human–computer interaction is to allow users to freely control the device with some simple operations. The human–computer interaction techniques include face recognition, language recognition, text recognition, and so on. As one of the important and powerful interaction methods, gesture recognition has attracted wide attention and been used in various fields, such as the video game industry, food industry, and machinery industry

Gesture recognition is a technique that is used to understand and analyze the human body language and interact with the user accordingly. This in turn helps in building a bridge between the machine and the user to communicate with each other. Gesture recognition is useful in processing information that cannot be conveyed through speech or text. Gestures are the simplest means of communicating something meaningful. Research into hand gestures has become an exciting and relevant field; it offers a means of natural interaction and reduces the cost of using sensors in terms of data gloves. As conventional interactive methods depend on different devices such as a mouse, keyboard, touch screen, joystick for gaming, and consoles for machine controls.

1.2 Main Objective

This thesis aims to implement a hand gesture recognition system to recognize several dynamic hand gestures using a camera so that appropriate actions can be taken accordingly. This system is created to control the music player system in a car. This gesture control system reduces the probability of causing a distraction-related accident by minimizing the need to take one's eyes away from the road to press the buttons on the music controller.

1.3 Structure of thesis

This thesis is divided into 5 chapters. In chapter 2, we mentioned some previous related works. Chapter 3 contains some information about the process pipeline of the system and algorithms used and chapter 4 shows the implementation of the system and the result. Finally, we added a conclusion to the results we achieved and our recommendations for future work in chapter 5 and 6 .

Chapter 2

Background

Gesture recognition is mainly divided into two types, Data Glove based and Vision-based approaches. The Data-Glove-based methods use sensor devices for capturing hand and finger motion data as input. The Vision-based methods require only a camera without the use of any extra devices.

2.1 Data Glove

Data glove in essence is a wired interface with certain tactile or other sensory units that were attached to the fingers or joints of the glove, worn by the user. Different types of sensors are used on active gloves that include flexible tubes with light, capacitive electrodes, flex sensors, and magnetic sensors to detect the curl of the finger. Accelerometer, Gyroscope, and other tilt sensors are used for the orientation and position of the hand, and proximity and touch sensors for the detection of contacts.

Mehdi et al [6] used gloves of 5DT (Dimensional Technologies), containing seven flex sensors. Five flex sensors were mounted on each of the fingers, one sensor was used to measure the tilt of the hand and one for the rotation of the hand. Artificial Neural Network (ANN) was used for classification, and 88% accuracy was obtained.

Joyeeta and Karen [8] used DG5 VHand 2.0 data gloves for hand gesture recognition. The features like the position of fingers and the hand were given by the gloves after which k-Nearest Neighbours (KNN) classifier algorithm was used.

Weissmann and Salomon [9] used Cyberglove that took into account angles made by 18 joints of the hand. Features extracted using this glove were angle made between the neighboring fingers, wrist pitch, thumb rotation which was then trained using Artificial Neural Network (ANN).

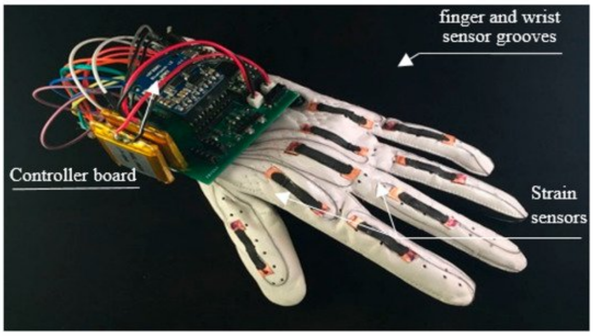


Figure 2.1: Sensor-based data glove

2.2 The Vision-based methods

Using computer vision techniques to identify gestures. Where the user performs specific gestures by single or both hands in front of the camera which connects with the system framework that involves different possible techniques that will be discussed in the following sections that extract features and classify hand gestures to be able to control some possible applications.

2.2.1 Colored Gloves or Markers

This method uses a camera to track the movement of the hand using a glove with different color marks. The colors on the glove enable the camera sensor to track and detect the location of the palm and fingers, which allows for the extraction of a geometric model of the shape of the hand. The advantages of this method are its simplicity of use and low price compared with the sensor data glove. However, it still requires the wearing of colored gloves and limits the degree of natural and spontaneous interaction.

Sayem et al [7] used two colored markers (red and green) worn on tips of the fingers to generate desired hand gestures and for marker detection and tracking used template matching with Kalman filter. The author tested the gestures of cursor move, left-click, right-click, zoom in, zoom out, Forward, and backward. overall average accuracy is 79.4%

Figure 2.2: Color-based recognition using glove marker

2.2.2 Color-Based Recognition of Skin Color

Skin color detection has been achieved using two methods. The first method is pixel-based skin detection, in which each pixel in an image is classified into the skin or not, individually from its neighbor. The second method is region skin detection, in which the skin pixels are spatially processed based on information such as intensity and texture. Color space can be used as a mathematical model to represent image color information. Several color spaces can be used including RGB, normalized RGB, HSV, YCrCb, YUV.

Chenglong et al [11] used YCbCr color model to distinguish skin-colored pixels from the background. The required portion of the hand was extracted using this color model and filtered using median filter and smoothing filter. The edges were detected and features extracted were hand perimeter, aspect ratio, hand area after which Artificial Neural Network (ANN) was used as a classifier to recognize a gesture. The accuracy rate obtained was 97.4%

RGB input image segmented hand

Figure 2.3: Color-based recognition of skin color

2.2.3 Skeleton-Based Recognition

The skeleton-based recognition specifies model parameters that can improve the detection of complex features. Where the various representations of skeleton data for the hand model can be used for classification, it describes geometric attributes and constraints and easily translates features and correlations of data, to focus on geometric and statistic features. The most common feature used is the joint orientation, the space between joints, the skeletal joint location, and the degree of angle between joints and trajectories and curvature of the joints.

Chenxuan et al [10] introduced hand segmentation using the depth sensor of the Kinect camera, followed by the location of the fingertips using 3D connections, Euclidean distance, and geodesic distance over hand skeleton pixels.

Devineau et al [4] used deep learning model using parallel Convolutional Neural Networks (CNN) to process hand skeleton joints' positions. The hand skeleton data returned by the Intel RealSense camera that is used contain 22 joints. 91.28% classification accuracy for the 14 gesture classes case and an 84.35% classification accuracy for the 28 gesture classes case was achieved.

De Smedt et al [3] used dynamic hand gestures using depth and skeletal dataset for a skeleton-based approach, where supervised learning (SVM) was used for classification with a linear kernel and Intel RealSense camera is also is used to obtain skeleton data. The hand skeleton-based method has demonstrated superior results of 88.24% and 81.90% of accuracy respectively for 14 and 28 different gestures.

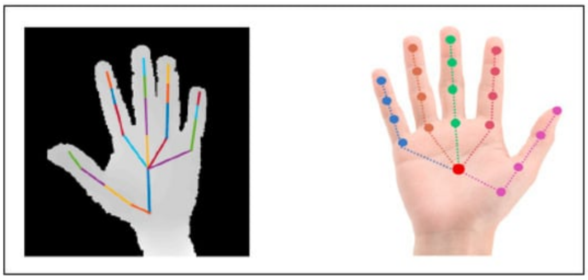


Figure 2.4: Skeleton recognition

2.2.4 Appearance-Based Recognition

This method depends on extracting the image features to model visual appearance such as hand and comparing these parameters with features extracted from the input image frames. The characteristic shape of hands has been utilized to detect them in images in multiple ways

Licsár and Szirányi [5] used a simple background subtraction method for hand segmentation and extended it to handle background changes to face some challenges such as skin-like color and complex and dynamic background then used boundary-based method to classify hand gestures.

Zhou et al [12] extracted the fingers where the edges were extracted from the gesture images by proposed novel algorithm.The hand posture is segmented and described based on the finger positions, palm center location, and wrist position. A weighted radial projection algorithm with the origin at the wrist position is applied to localize each finger.

Figure 2.5: Appearance-based recognition method

2.2.5 Deep-Learning Based Recognition

Artificial intelligence offers a good and reliable technique used in a wide range of modern applications because of using a learning role principle. Deep learning used multilayers for learning data and gives a good prediction result. The most challenge facing this technique is the required dataset to learn algorithm which may affect time processing.

Alnaim et al [1] proposed seven popular hand gestures captured by mobile camera and generate 24,698 image frames. The feature extraction and Adapted Deep Convolutional Neural Network (ADCNN) were utilized for hand classification. The experiment evaluates results for the training data 100% and testing data 99%, with execution time 15,598s.

Bao et al. [2] used a method based on a deep Convolutional Neural Network (CNN), where the resized image directly feds into the network ignoring segmentation and detection stages to classify hand gestures directly. The system works in real-time and gives a result with a simple background of 97.1% and with a complex background of 85.3%.

Chapter 3

Methodology

In our gesture recognition system, we have included a total of six hand gestures, where two of them are static gestures and four is a dynamic gesture. The gesture recognition is structured following the computer vision techniques of skeleton-based recognition. The system was developed using OpenCV module in python, which is a huge open-source library for computer vision, machine learning, and image processing. With OpenCV, we can capture a video and obtain the image frames. The output frames go through two steps hand detection and gesture recognizer.

3.1 Hand detection

MediaPipe is used to detect the hand in the images captured by the camera and perceive the shape and motion of hands. MediaPipe is an open-source framework designed by Google, which allows building pipelines to perform inference over arbitrary sensory data. It has a selection of high accuracy machine learning models for human body parts detection and tracking, able to track key points on these body parts, which are called landmarks. Google trained these custom-built models on its largest and most diversified datasets. In particular, MediaPipe comes with a hand tracking solution named MediaPipe Hands

MediaPipe Hands is a high-fidelity hand and finger tracking solution and produces hand landmark model that outputs precise landmark localization of 21 3D hand-knuckle coordinates inside the detected hand regions. Each landmark is composed of x, y, and z. x and y are normalized to [0.0, 1.0] by the image width and height respectively.z represents the landmark depth with the depth at the wrist being the origin, and the smaller the value the closer the landmark is to the camera.

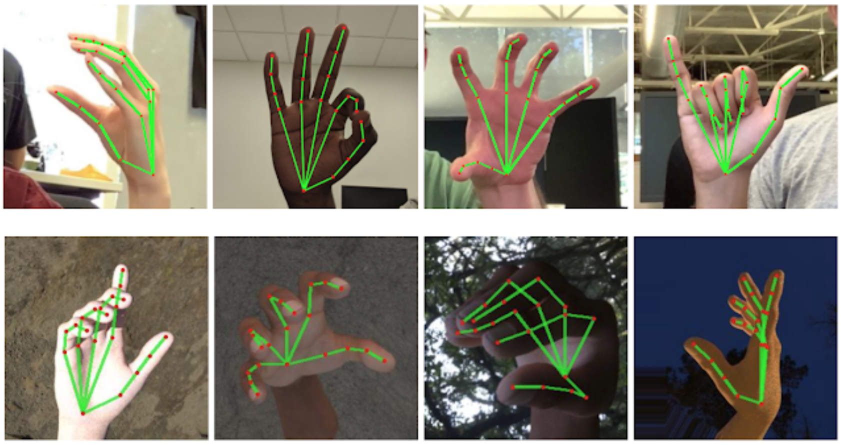


Figure 3.1: The hand landmark models output when the hand is detected

3.2 Gesture Recognizer

The inputs to the Gesture Recognizer are the 21 3D landmarks generated by MediaPipe. Each corresponding to a point on the hand and each keypoints consists of three coordinates (x,y,z). The gesture Recognizer system works by understanding hand gestures that are recognized from the hand poses and their movements.

3.2.1 Hand Poses Recognition

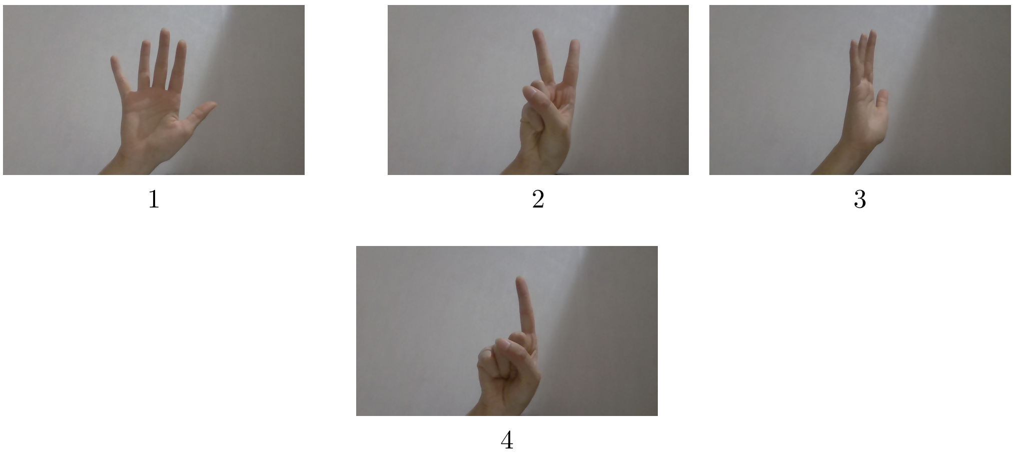


Figure 3.2: Hand poses to be recognized

An algorithm is used to recognize the hand poses as shown in Fig 3.2 by determining the state of each finger either open or close and the orientation state of the hand either perpendicular or parallel.

Algorithm 1 hand poses states

if $landmark[3].x < landmark[2].y$ **and** $landmark[4].y < landmark[2].y$ **then**
  $Thumbstate$ ←open
else
  $Thumbstate$ ←close
end if
if $landmark[7].y < landmark[6].y$ **and** $landmark[8].y < landmark[6].y$ **then**
  $FirstFingerstate$ ←open
else
  $FirstFingerstate$ ←close
end if
if $landmark[12].y < landmark[10].y$ **and** $landmark[11].y < landmark[10].y$ **then**
  $SecondFingerstate$ ←open
else
  $SecondFingerstate$ ←close
end if
if $landmark[16].y < landmark[14].y$ **and** $landmark[15].y < landmark[14].y$ **then**
  $ThirdFingerstate$ ←open
else
  $ThirdFingerstate$ ←close
end if
if $landmark[20].y < landmark[18].y$ **and** $landmark[19].y < landmark[20].y$ **then**
  $ThirdFingerstate$ ←open
else
  $FourthFingerstate$ ←close
end if
if $landmark[5].x = landmark[9].x$ **and** $landmark[5].x = landmark[13].x$ **and** $landmark[5].x = landmark[17].x$ **and** $landmark[9].x = landmark[13].x$ **and** $landmark[9].x = landmark[17].x$ **and** $landmark[13].x = landmark[17].x$ **then**
  $Orientationstate$ ←perpendicular
else
  $Orientationstate$ ←parallel
end if

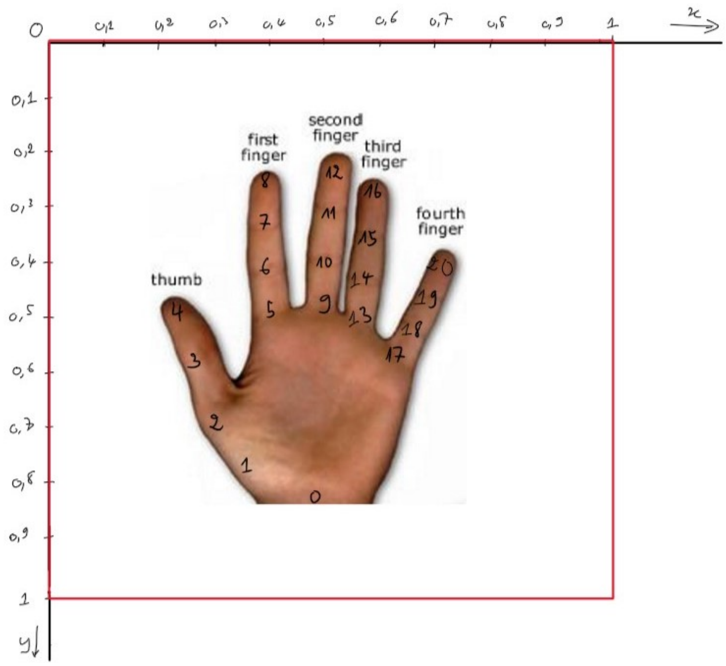


Figure 3.3: Hand landmark

To recognize four basic hand poses, we derived a set of recognition rules. The set of six features from the states of all fingers and state of orientation was used to decode the meaning of the four hand poses. The derived recognition rules are given in Table 3.1.

Thumb	First Finger	Second Finger	Third Finger	Fourth Finger	Orientation	Pose
open	open	open	open	open	parallel	1
close	open	open	close	close	parallel	2
open	open	open	open	open	perpendicular	3
close	___	close	close	close	parallel	4

Table 3.1: Recognition rules of hand poses based on the states

As shown in Table 3.1, First Finger state of pose 4 is not defined .The reason is that the close state of the fingers is detected when the fingers are bend, however, the first finger needs to be straight or bent as it will be moving and it will be discussed with more details in the upcoming sections. A modification on the state of the first finger is made as if the y value of landmark 7 and the y value of landmark 8 are less than y value of landmark 5 then the finger is straight or bent else it is completely closed. Pose 4 will be recognized when the state of the first finger is in a bent or straight state.

The proposed algorithm to determine perpendicular orientation state is not that accurate as the hand could not be always straight as it could be slightly oriented to the left or right. To improve the hand pose detection modification should be applied, x value of landmark 17 is the reference position point which a certain value is added and subtracted from it to create a left and right threshold that x values of landmark 5, landmark 9, and landmark 13 do no exceed.

Algorithm 2 orientation hand algorithm modification

$RightThreshold \leftarrow landmark[17].x + RightThresholdValue$
$LeftThreshold \leftarrow landmark[17].x - LeftThresholdValue$
if $landmark[5].x < RightThreshold$ **and** $landmark[5].x > LeftThreshold$ **and** $landmark[9].x < RightThreshold$ **and** $landmark[9].x > LeftThreshold$ **and** $landmark[13].x < RightThreshold$ **and** $landmark[13].x > LeftThreshold$ **then**
  $Orientationstate \leftarrow$perpendicular
else
  $Orientationstate \leftarrow$parallel
end if

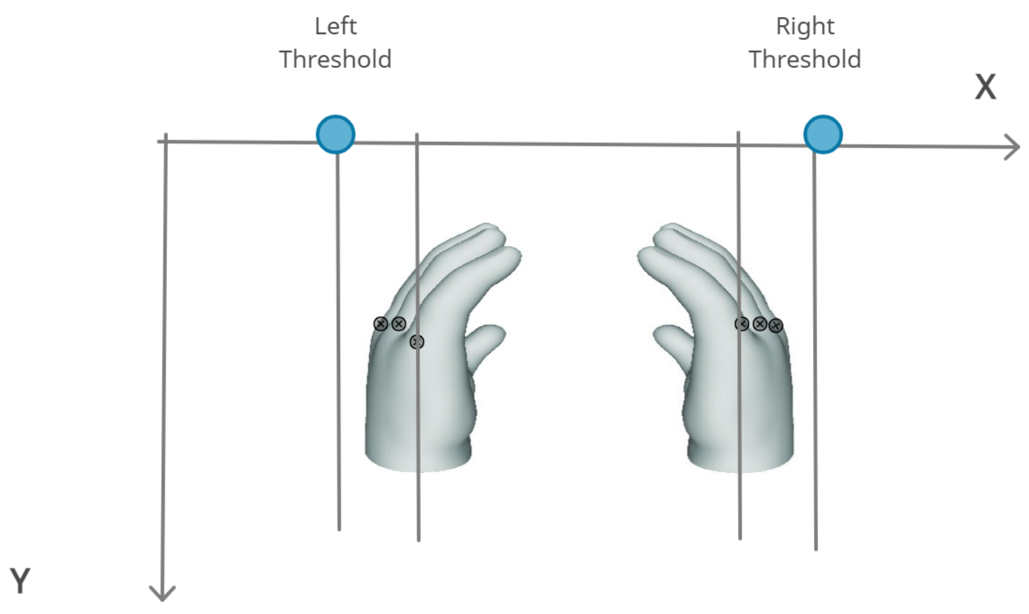


Figure 3.4: Perpendicular hand state

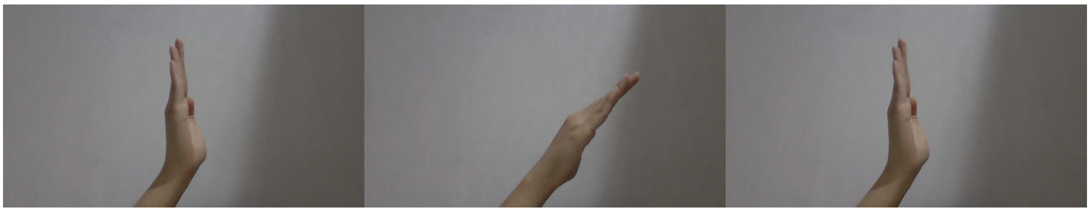
3.2.2 Static Hand Gestures Recognition

Static gestures are gestures that can be described by a single hand pose. To understand Gestures 1 and 2, our system recognizes Pose 1 and 2 respectively. And it is only recognized if it is in a static state which is the hand pose is not moving. The static state is detected if the x and y values of landmark 0 do not change or move for a consecutive number of frames threshold. However, hand motion is jerky and unstable at times so the movement of point could change as the displacement does not exceed a certain threshold.

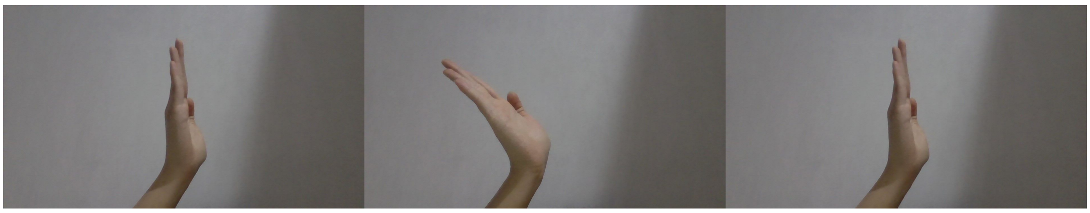
3.2.3 Dynamic Hand Gestures Recognition

Dynamic hand gesture is making movements either using the whole hand or just the fingers. After recognizing the hand poses as explained, movement is recognized by tracking their positions. Tracking points is compared between the previous and current frame to determine the change and the state of movement.

As shown in fig 3.5, movement A of pose 3 starts as the hand is stable and does not move, it moves to the right except for the wrist of the hand then it moves to the left and it stops as the hand is stable. movement B starts as the hand is stable and does not move, it moves to the left except for the wrist of the hand then it moves to the right and it stops as the hand is stable.



A



B

Figure 3.5: Movements of pose 3

Gestures 3 and 4 can be understood by recognizing Pose 3 and then tracking the x values of landmark 17, landmark 18, landmark 19, and landmark 20 points. The proposed algorithm is used to determine the state of the movements of tracking points.

Algorithm 3 State of tracking points of pose 3

while $pose = pose3$ **do**
  $trackingpoints \leftarrow landmark[17].x, landmark[19].xandlandmark[20].x$
  if $previousframe(tackingpoints) < currentframe(tackingpoints)$ **then**
    $state \leftarrow right$
  else
    if $previousframe(tackingpoints) > currentframe(tackingpoints)$ **then**
      $state \leftarrow left$
    else
      $state \leftarrow stable$
    end if
  end if
  state is inserted in the sequence
end while

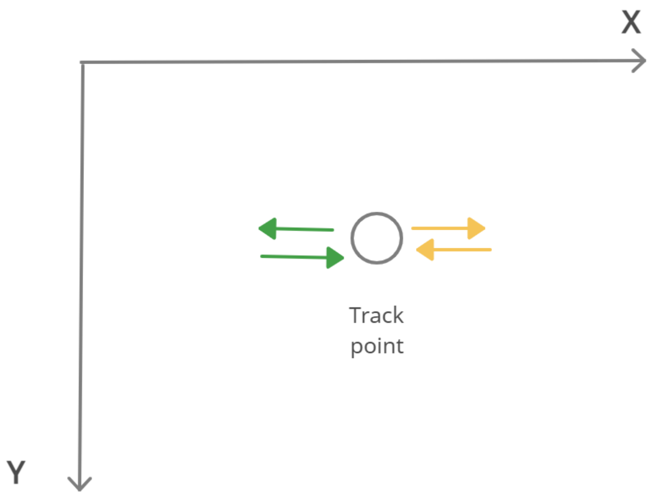


Figure 3.6: Tacking point

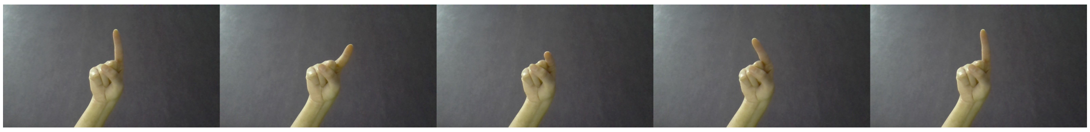
To recognize the movement of pose 3, we derived a set of recognition rules. The sequence of the states of tracking points was used to decode which movement it is. The derived recognition rules are given in Table 3.2

Movement	Sequence
A	Stable → Right → Left → Stable
B	Stable → Left → Right → Stable

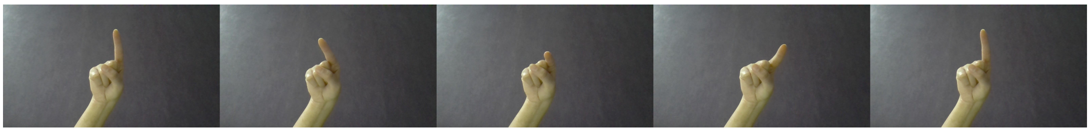
Table 3.2: Recognition rules of movements based on the states sequence

The proposed tracker algorithm is quite not accurate and could lead to wrong recognition as hand motion is jerky and unstable at times, so some improvements were introduced to improve the tracker. In the Right or Decreasing state, the change of the values of tacking points between the previous and current frame exceeds the displacement threshold. As the transition between Increasing and Decreasing state or between Increasing and Decreasing state could not be fast or there is a little pause in between and stable state could be detected messing up the sequence, so the stable state is determined when tracking points is not in Increasing or Decreasing state for a consecutive number of frames threshold. Also, The static state of the wrist is detected if the x and y values of landmark 0 as the movement of point could change as the displacement does not exceed a certain threshold for a consecutive number of frames threshold.

Gestures 5 and 6 can be understood by recognizing Pose 4 and then tracking the x and y values of landmark 8 points. As shown in fig 3.5, movement C shows the first finger moving in a circular clockwise motion. movement B shows the first finger moving in a circular anti-clockwise motion. In both movements the wrist of the hand is stable.



c



d

Figure 3.7: Movements of pose 4

The proposed algorithm is used is to collect tracking points of every frame to detect if points fit a circle. The tracking points are drawn on canvas by connecting the points.

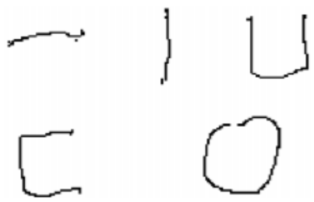


Figure 3.8: Example of drawings on canvas

To detect circular shape OpenCV findContours() function is used to detect the contours in canvas. contours are mathematical structures with shapes that are formed by joining the points covering an area of similar intensity or color. The contours are a useful tool for shape analysis and object detection and recognition. Then approxPolyDP() function is used to find what kind of polygon is the contour gives the output as a number of vertices for the polygon. For eg., a square contour will give the output as 4 points, pentagon as 5 points. Circular objects will have a higher number of points. Here, we see that depending on the type of objects in the image, polygons with greater than 8 vertices form curvier shapes, circles, and ellipses.

Figure 3.9: Example of contour of a shape

To find if the movement is clockwise or anti-clockwise , we apply he signed area of non-self intersecting polygon which is

$$A = \frac{1}{2}(x_1y_2 - x_2y_1 + x_2y_3 - x_3y_2 + ... + x_ny_1 - x_1y_n)$$

The input coordinates to the area formula are the tracking points. The area of the detected polygon is defined to be positive if the points are arranged in clockwise order, and negative if they are in anti-clockwise order

Chapter 4

Implementation and Results

4.1 Implementation

An interface hand gesture interface system is implemented and can visually render the result of the recognition of the gesture.

All the results obtained will be under the following assumptions:-

1. The user should be in an illuminated with a uniform light source place
2. The user should keep his hand stationary when the re-initialization takes place (new gesture to execute)
3. While tracking the user should avoid sudden or very fast movements and should not rotate the hand too much
4. The user should be close to the camera

The interface consists of three windows. The first window consists of the video input that is captured from the camera with the corresponding name of the gesture and pose detected. The second window displays the drawing canvas for gesture 5 and 6 circle contour detection, the canvas is erased when circle contour is detected. The advantage of adding the video and drawing window as a part interface is to make the user aware of the background inconsistencies that would affect the input to the system and thus they can adjust their laptop or desktop web camera to avoid them. This would result in better performance. The third window is a media player and it is built with VLC and PySimpleGUI which are both libraries offered by Python.PySimpleGUI is used to create GUIs and VLC is used to create playlists of local or online streaming media and control playlists.

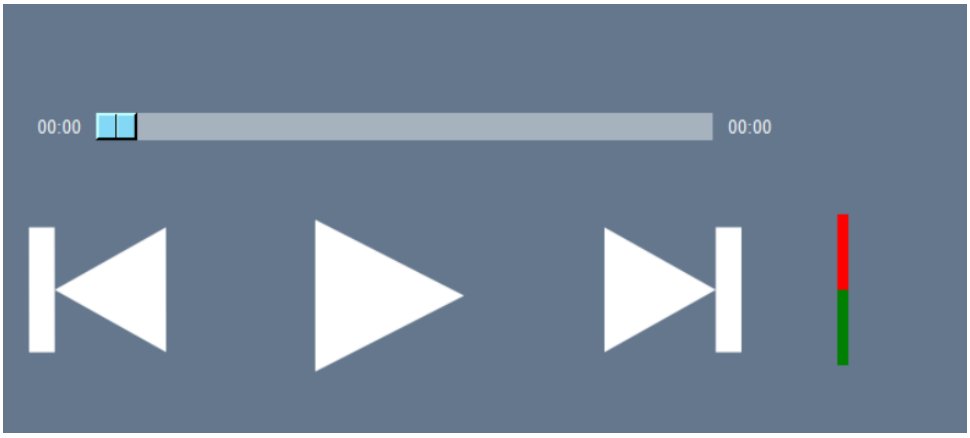


Figure 4.1: media player graphic user interface

Gesture	Functionality
1	Play
2	Pause
3	next
4	previous
5	volume up
6	volume down

Table 4.1: Functionality of gestures

Gesture 1 activates the system and plays the first song on the playlist or current song if it is paused. The media player graphic interface change as play button change to pause button and song name appears. Gesture 2 to pause the current playing song, the media player graphic interface change as pause button change to play button. Gesture 3 to skip the current playing song to play the next song. The media player graphic interface as 3 skip button changes color and flashes with green color than the song name change. Gesture 4 to skip the current playing song to play the previous song. The media player graphic interface change as the previous skip button changes in color and flashes with green color than the song name change.Gesture 5 to increase the volume of the song and the green bar height in the media player graphic interface increase. Gesture 6 to decrease the volume of the song and the green bar height in the media player graphic interface decrease.

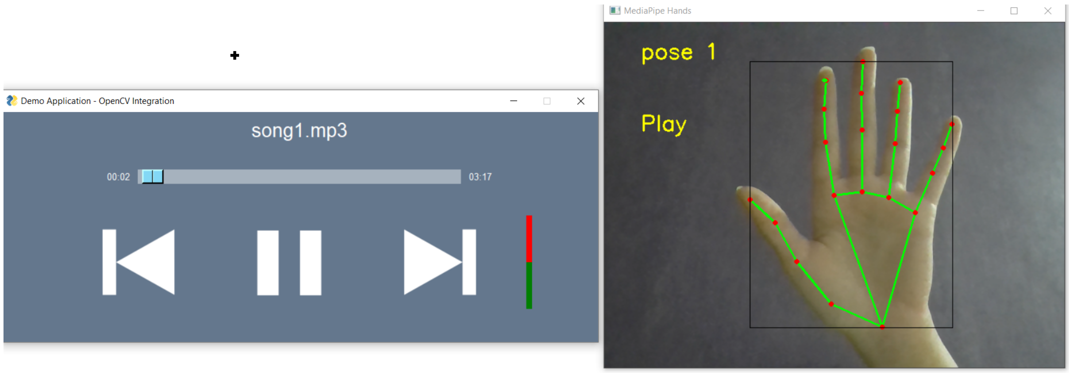


Figure 4.2: Gesture 1 recognition

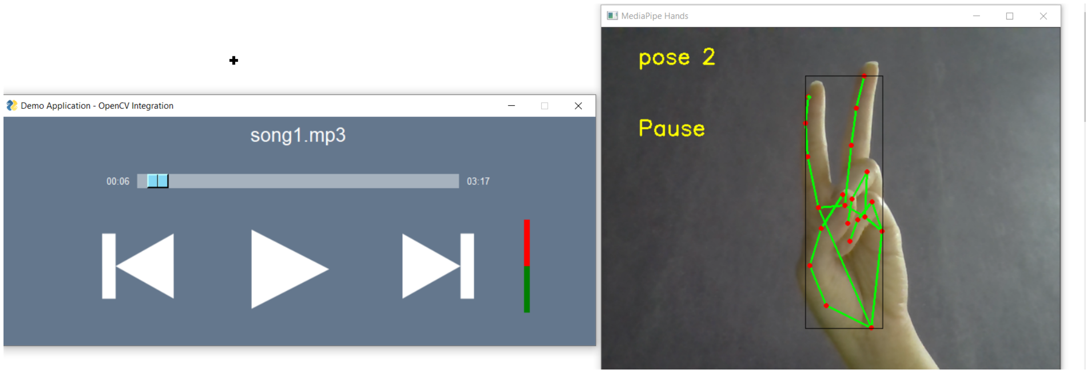


Figure 4.3: Gesture 2 recognition

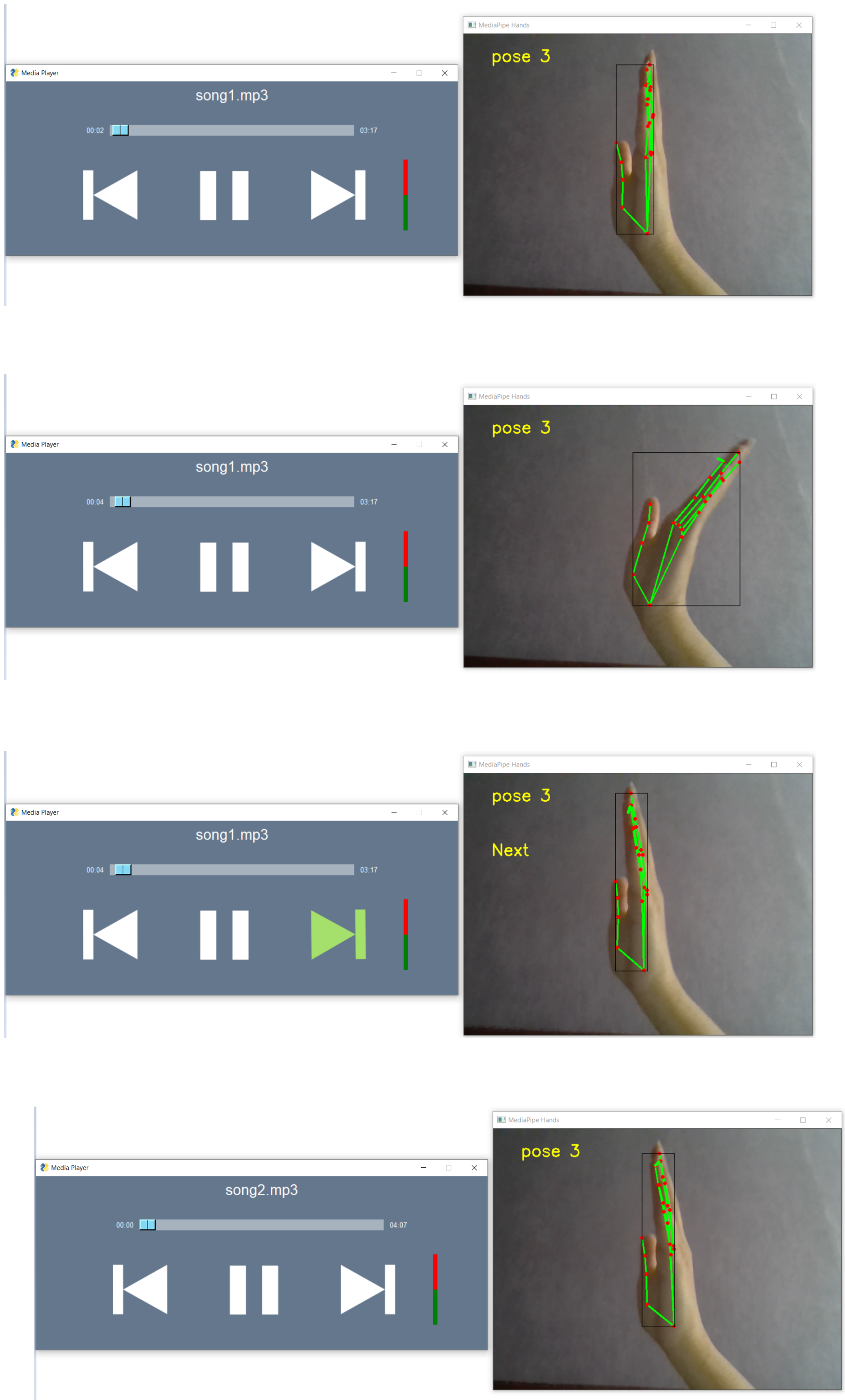


Figure 4.4: Gesture 3 recognition

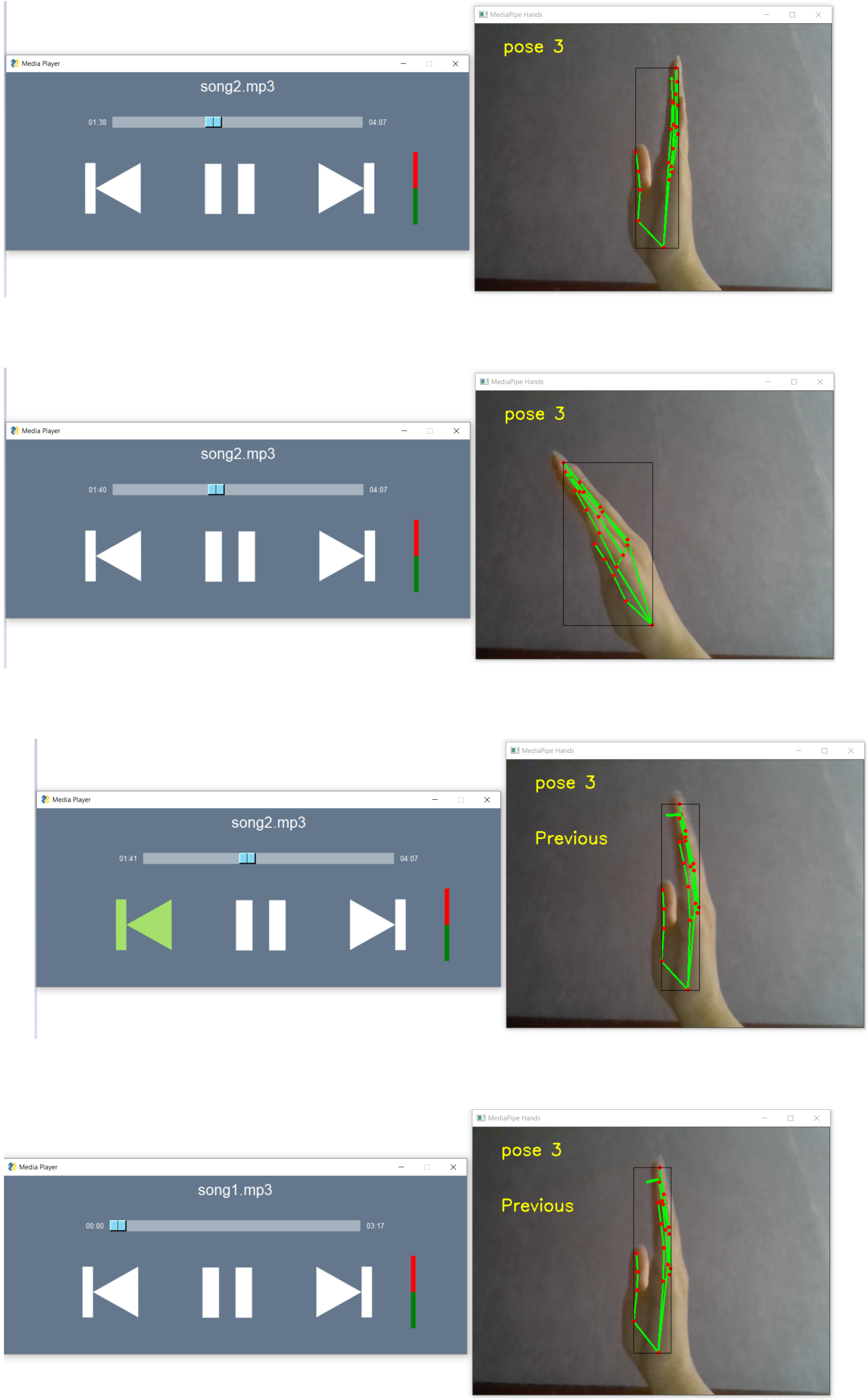


Figure 4.5: Gesture 4 recognition

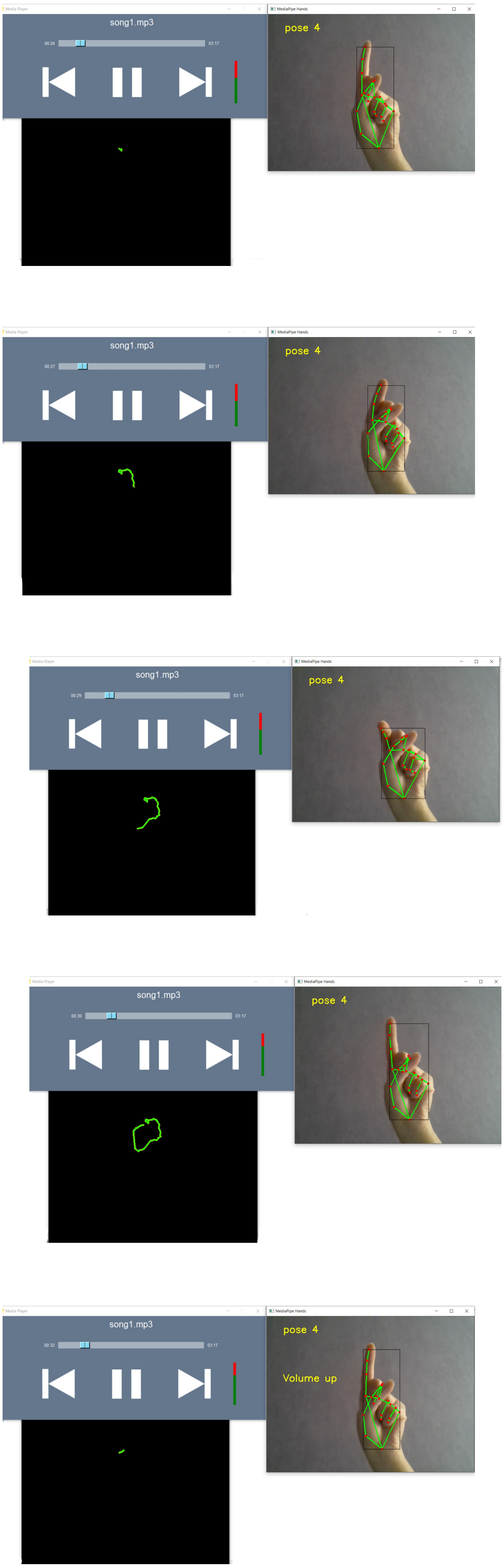


Figure 4.6: Gesture 5 recognition

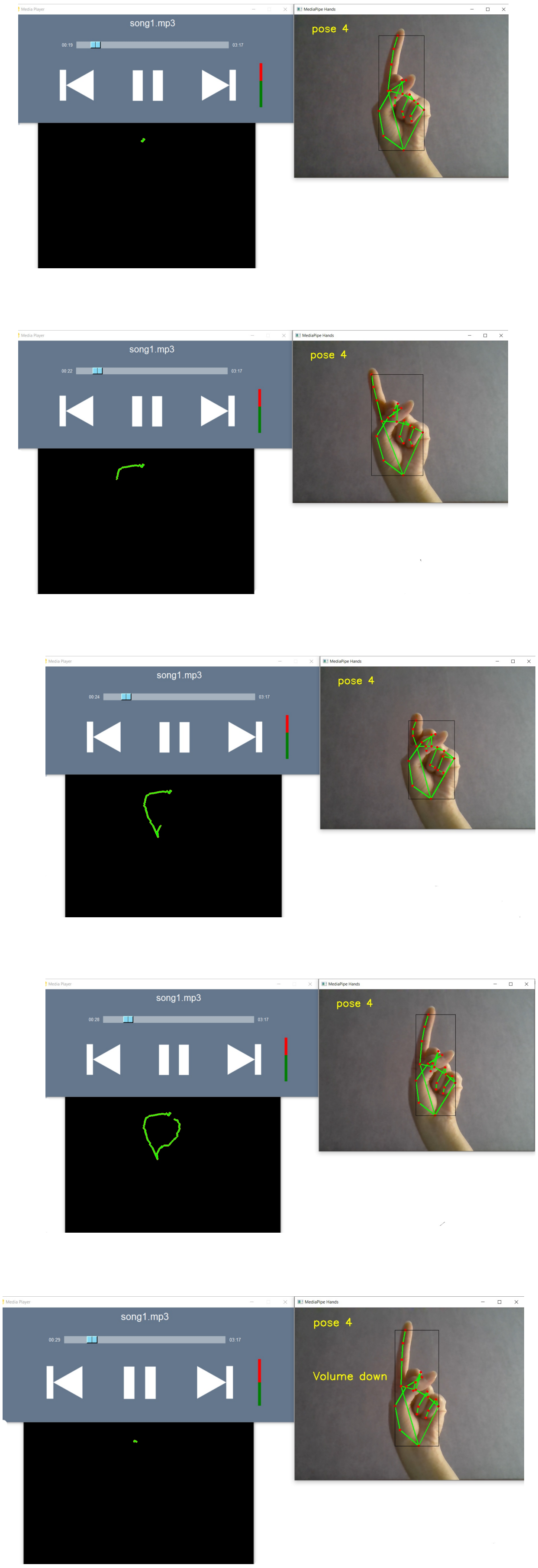


Figure 4.7: Gesture 6 recognition

4.2 Experimental Results

To evaluate our hand gesture recognition system, a test was conducted by 4 participants. The four participants performed the six gestures 20 times randomly satisfying the assumption stated earlier in the paper. The average recognition rate is 87.7%.

Participant	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
1	20/20	20/20	19/20	17/20	16/20	15/20
2	20/20	20/20	18/20	17/20	15/20	15/20
3	20/20	20/20	18/20	16/20	17/20	14/20
4	20/20	20/20	17/20	18/20	14/20	15/20

Table 4.2: Experimental Results

The table above 4.2 illustrates the number of correctly detected recognition in each gesture out of the 20 times.

Chapter 5

Conclusion

We were able to create a gesture recognition system that did not utilize any markers or gloves using a camera, hence making it more user-friendly and low cost. In this gesture recognition system, we have aimed to provide gestures, covering almost all aspects of human interaction with car music player system functionalities such as, playing, pausing, skipping, increasing, and decreasing the volume of the song. we detected hand using MediaPipe and preformed various algorithms to recognize hand gesture by recognizing hand poses and tracking it. We have achieved the recognition accuracy of 87.7% over the six hand gestures from four subjects

Chapter 6

Future Work

The major extension to this work is to add more gestures to implement more functions. Despite the algorithm efficiency and robustness, some parts need enhancing in the future and some extra features could be added to the project. the feature should be added to change the threshold depending on how far or close to the camera, as in the implemented system depend on a fixed threshold based that is closer to the camera. To increase the accuracy of the recognition system, an alternative approach could be implemented to track circular clockwise or anticlockwise movements of gestures 5 and 6 as they hold the least accuracy rate. Also, to increase hand detection accuracy as MediaPipe could detect skeleton-like shapes in the background, background subtraction methods could be approached. In the future, more participants should engage in experimental testing for more data collection.

Appendix

Appendix A

Lists

List of Figures

List of Tables

Bibliography

[1] Norah Alnaim, Maysam Abbod, and Abdulrahman Albar. Hand gesture recognition using convolutional neural network for people who have experienced a stroke. In *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 1–6, 2019.

[2] Peijun Bao, Ana Maqueda, Carlos Del-Blanco, and Narciso García. Tiny hand gesture recognition without localization via a deep convolutional network. *IEEE Transactions on Consumer Electronics*, 63:251–257, 08 2017.

[3] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, Joris Guerry, Bertrand Le Saux, and David Filliat. SHREC'17 Track: 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset. In I. Pratikakis, F. Dupont, and M. Ovsjanikov, editors, *3DOR - 10th Eurographics Workshop on 3D Object Retrieval*, pages 1–6, Lyon, France, April 2017.

[4] Guillaume Devineau, Fabien Moutarde, Wang Xi, and Jie Yang. Deep learning for hand gesture recognition on skeletal data. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, pages 106–113, 2018.

[5] Attila Licsár and Tamás Szirányi. User-adaptive hand gesture recognition system with interactive training. *Image and Vision Computing*, 23:1102–1114, 11 2005.

[6] Syed Atif Mehdi and Yasir Niaz Khan. Sign language recognition using sensor gloves. pages 2204 – 2206 vol.5, 12 2002.

[7] Sayem Siam, Jahidul Sakel, and Md Kabir. Human computer interaction using marker based hand gesture recognition. 06 2016.

[8] Joyeeta Singha and Karen Das. Hand gesture recognition based on karhunen-loeve transform. *Mobile and Embedded Technology International Conference (MECON)*, 06 2013.

[9] J. Weissmann and R. Salomon. Gesture recognition for virtual reality applications using data gloves and neural networks. *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No.99CH36339)*, 3:2043–2046 vol.3, 1999.

[10] Chenxuan Xi, Jianxin Chen, Chenxue Zhao, Qicheng Pei, and Lizheng Liu. Real-time hand tracking using kinect. pages 37–42, 02 2018.

[11] Chenglong Yu, Xuan Wang, Hejiao Huang, Jianping Shen, and Kun Wu. Vision-based hand gesture recognition using combinational features. In *2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, pages 543–546, 2010.

[12] Yimin Zhou, Guolai Jiang, and Yaorong Lin. A novel finger and hand pose estimation technique for real-time hand gesture recognition. *Pattern Recognition*, 49:102–114, 2016.