Project Report

Hand Gestures-based Media Controls

Winter Semester 2022-23

Submitted By:

Shivaji Annem , 20MIC0091

Palepu Venkata Hemanth, 20MIC0105

Under the guidance of : ANISHA M.LAL

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

ABSTRACT

This report presents a novel approach in Human-Computer Interaction, focusing on controlling media playback through hand gestures. The objective is to design and implement a prototype system that enables users to manage media playback on a computer intuitively and efficiently, without requiring a physical controller. The final report addresses existing challenges, such as dependency on physical interfaces and limited mobility. The proposed solutionutilizes computer vision and image processing techniques to recognize and interpret hand gestures, allowing users to execute commands like play, pause, and volume adjustments through hand movements. The system is implementedusing Python and OpenCV, demonstrating a promising advancement in user interaction with media playback systems..

This approach leverages **image processing techniques** within the domain of Human-Computer Interaction to interpret and respond to hand gestures. By employing computer vision algorithms, the system recognizes specific gestures, enabling users to control media playback seamlessly. This integrationof image processing enhances the system's ability to accurately interpret and respond to user inputs.

OBJECTIVES

The objectives are as follows:

1. To design and implement a prototype system that allows users to control media playback using hand gestures.
2. To evaluate the effectiveness and usability of the proposed system in comparison to traditional media playback controls.
3. To identify and address the issues with the existing media playback systems, including the need for physical controllers or interfaces, limited mobility, and lack of intuitive controls.
4. To explore the potential of hand gesture-based interfaces as a viable and practical solution for controlling media playback systems.
5. To provide insights and recommendations for future research and development in the field of Human-Computer Interaction.

MOTIVATION

It is becoming increasingly difficult for specially-abled people, specifically people who are deaf and dumb to interact with latest software without any formal knowledge of such systems. Such hand gestures-based software help them with interacting with a wide variety of media systems and can be the pioneer of disabled-friendliness while designing and configuring the functionalities of future applications.

With increasing improvement in technology, reaction time and ease of operations are the issues. Here is where human-computer interaction comes into play. This interplay is unrestricted and challenges the used gadgets consisting of the keyboard and mouse for input. Gesture recognition has been gaining tons of attention. Gestures are instinctive and are often utilized in everyday interactions. Therefore, communicating using gestures with computer systems creates an entire new trend of interaction.

LITERATURE SURVEY

SNo.	Paper Details	Methods Used	Merits	Demerits
1.	G. Krastev and I. Ralev, "Gesture Based System for User Interface Control," 2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, IEEE 2021. [1]	The paper researches the features of the Creative Intel RealSense SR300 camera as a means of hand gesture recognition and using them to control certain application software. For this purpose, is created console application in JAVA operate the camera and hand gesture recognition that used to control the media player, photo view, change zoom and regime change maps in web-based version of Google Maps and play or pause clips, enter or exit of	Useful for Google Maps and YouTube applications.	Only certain features have been recognized using hand-based gestures and cannot be applied in general to all the media-based applications.

		YouTube full screen mode.		
2.	Monisha Sampath, Priyadarshini Velraj, Vaishnavii Raghavendran, and M Sumithra, “Controlling media player using hand gestures with VLC media player,” World Journal of Advanced Research and Reviews, vol. 14, no. 3, pp. 466–472, Jun. 2022. [2]	The design for the proposed device is to stumble on hand gestures and control the media participant without touching the keyboard. PyAutogui is used to configure the media player with the python code.	Works with VLC Media Player.	Consists of only 5 hand gestures.
3.	Bilvika, K & K, Sneha & M, Sahana & Patil, Tejaswini. (2021). “Face and Hand Gesture Recognition System for Controlling VLC Media Player”. International Journal of Scientific Research in Science and Technology. [3]	This gesture-based interaction controls the functions of a media player through webcam. It has been achieved through Image Processing and Pyautogui. The image processing by using OpenCV and for further tasks like	More functionalities and associated with webcam.	Latency.

		Gaussian and Morphological transformation.		
4.	Sakshi Shinde, Sarthak Mushrif, Aditya Pardeshi, Dhairyasheel Jagtap, Prof. Vandana Rupnar. “Gesture-based Media Player Controller”. International Journal of Research Publication and Reviews, May 2022. [4]	In this paper, an application has been designed for personal computer interaction that uses a variety of computer recognition techniques to detect hand gestures to control a VLC media-player. The purpose and objectives of this app are to use the free interface of a natural device, which sees hand gestures as commands. The app uses a webcam used to capture images. To control a VLC media player using a defined touch, the app focuses on a specific VLC function that is widely used.	Works with VLC Media Player.	Very less functionalities. Uses obsolete techniques.

5.	P. Maloo, “Controlling Media Player with Hand Gestures,” International Journal of Engineering Research and Technology, vol. 11, no. 11, Nov. 2022, doi : 10.17577/IJERT 11IS110088. [5]	The project's stated goal is to use hand gestures to operate standard media player controls. In this paper, a hand gesture recognition system which is able to control computer media players is offered. Hand gestures are one of the key elements to interact with the smart system. The hand gestures are used in recognition mechanism of a computer media player, for instance, volume down/up, next music, etc.	Uses convexity hull and convexity defects to accurately pinpoint finger-tips.	Only works well with plain background.
6.	G. D. Nagalapuram, R. S, V. D, D. D and D. J. Nazareth, "Controlling Media Player with	The proposed web application enables the user to use their local	It increases efficiency and makes interaction effortless by letting the	High misclassification rate.

	Hand Gestures using Convolutional Neural Network," 2021 IEEE Mysore Sub Section International Conference (MysuruCon), Hassan, India, 2021, pp. 79-86, doi: 10.1109/ MysuruCon52639. .2021.9641567. [6]	device camera to identify their gesture and execute the control over the media player and similar applications (without any additional hardware). This involves using Deep Learning-based CNN which increases the efficiency.	user control his/her laptop/desktop from a distance.	
7.	K. Venkatasalam, Sourav, V. Yashwanth Venkata, S. Chowdary, and R. Chandu, "AUDIO PLAYER CONTROLLING BASED ON HAND GESTURE TECHNIQUE," 2604. International Research Journal of Modernization in Engineering Technology and Science, June, 2022.	This undertaking pursuits to increase an audio participant managed via way of means of human gestures the use of media pipe technology. It makes use of the webcam to seize the user's gestures and carry out primary operations	Usage of Media-pipe technique gives cross-platform advantage.	Suitable for audio platforms.

	[7]	including extent up and down, play and pause. Gestures act as direct instructions for movements including gambling or pausing the audio extent at the screen, primarily based totally at the user's gestures. You can use hand gestures to manipulate positive capabilities at the computer, including Play or pause sounds and growth and reduce extent primarily based totally on histogram values.		

ISSUES IN EXISTING SYSTEM

Most of the existing systems have either very high latency leading to user-unfriendly environment or have high rate of misclassification leading to less efficient model.

All of the existing models have very few hand gestures recognized by the system and has limited working functionality. The scope of hand gestures recognized by the system needs to be broadened which will be done by our proposed model.

PROPOSED SYSTEM

Overview of the Proposed System

Our model will segregate the screen into parts wherein one of the parts corresponds to actual hand gesture detection. This makes sure that the controls don’t change when the user makes hand gestures unknowingly and the controls don’t change when undesired for.

Our model will also provide features for adjusting hand saturation and hue values for proper detection and recognition of hand gestures in appropriate situation. This leads to efficient detection of hand contours.

The model will include a variety of hand gestures for a spectrum of functionalities ranging from pause/play to volume controls, and forward to rewind.

System Architecture

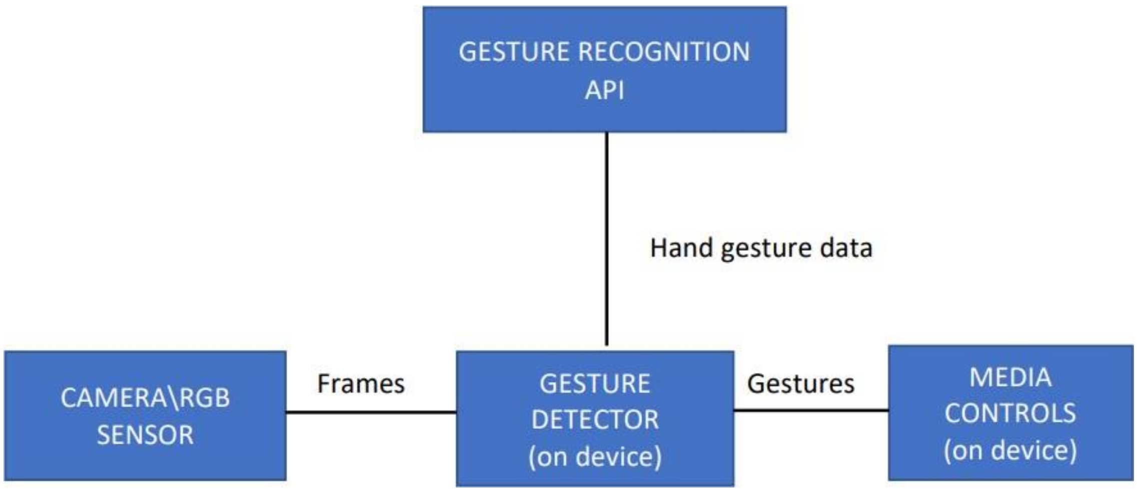
The system consists of the following components:

1. **Camera / RGB Sensor**: This component is responsible for capturing video data of the user's hand gestures. RGB sensors capture colour information, while depth sensors capture both colour and depth information. The type of sensor used depends on the application requirements and the level of detail needed for gesture recognition. For example, depth sensors like the Microsoft Kinect can provide more accurate 3D information for complex hand gestures.

2. **Gesture Recognition API**: This component analyzes the video data captured by the camera to recognize the user's hand gestures. This API typically uses machine learning algorithms or other pattern recognition techniques to detect and classify different hand gestures. For example, the API may use a convolutional neural network to analyze image frames and identify specific hand gestures based on their distinctive visual features.

3. **Gesture Detector (on device)**: This component receives the hand gesture data from the Gesture Recognition API and detects the specific gesture being performed by the user. This may involve filtering out any noise or unwanted data to improve the accuracy of gesture recognition. The Gesture Detector may also use algorithms to track the movement of the hand over time, allowing it to recognize more complex gestures that involve multiple hand movements.

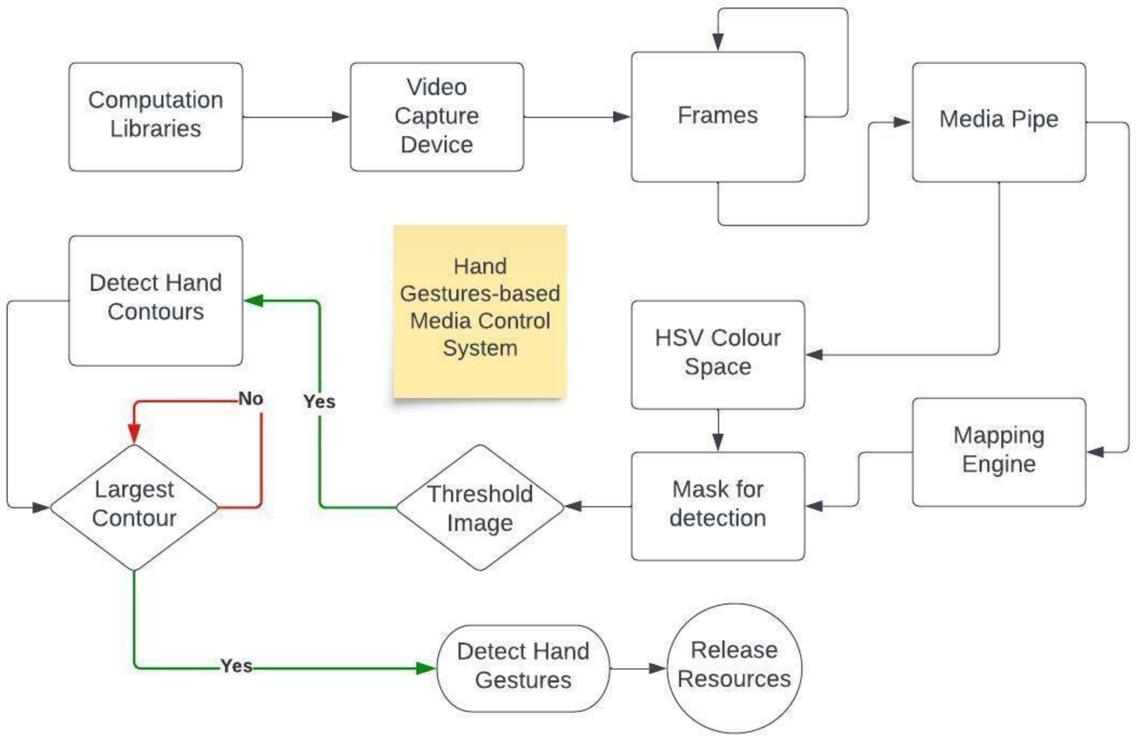
4. **Media Controls (on device)**: This component receives the detected gesture information from the Gesture Detector and performs the corresponding media control action. For example, if the user performs a swipe gesture to the left, the media control component may interpret this as a command to skip to the previous track in a music player app.



The system can be implemented on a single device like a smartphone or tablet, where the camera and all other components are integrated into the device. Alternatively, the system can be implemented on multiple devices, such as a camera and a separate media device, allowing for more flexibility and scalability in larger applications. Additionally, the Gesture Recognition API can be implemented as a cloud-based service, which allows the system to work across multiple devices and platforms, while also providing more robust and adaptable solutions, as it can be easily updated and integrated with different platforms and devices.

Overall, a hand-gestures based media control system involves multiple components that work together to provide an intuitive and seamless user experience. The system utilizes camera or sensor technology, machine learning algorithms, and gesture detection and media control components to recognize and respond to the user's hand gestures in real-time.

Functional Architecture



The system architecture follows the standard image processing pipeline for detecting hand gestures. The system first captures the video stream and extracts the ROI for processing. The ROI is converted to HSV colour space to improve colour detection and segmentation. The system then creates a mask to detect the hand region, and thresholds the image to filter out noise. The contours of the hand are detected and the largest contour is selected. The convex hull and defects of the contour are found to detect hand gestures. Finally, the system uses the PyAutoGUI library to control the mouse and keyboard based on the number of fingers detected.

Module-wise Architecture

1. ***Import necessary libraries:***

- Import OpenCV library for image processing
- Import NumPy library for numerical operations on the image
- Import Math library for mathematical calculations
- Import PyAutoGUI library for controlling mouse and keyboard
- Import Time library for delay operations

2. ***Capture the video stream:***

- Open a video capture device (webcam) for capturing video frames
- Create a while loop to continuously capture frames from the video stream
- Flip the frames horizontally to avoid mirror imaging
- Resize the frames to a standard size (600 x 500 pixels)
- Draw a rectangle on the frame to represent the region of interest (ROI) for detecting hand gestures
- Crop the frame to extract the ROI for processing

3. ***Convert the frame to HSV color space***:

- Convert the cropped frame to HSV color space for better color detection and segmentation
- Create trackbars for adjusting the lower and upper HSV color values to detect the skin color of the hand

4. ***Create a mask for detecting the hand:***

- Set the lower and upper HSV color bounds based on the trackbar values
- Create a mask by thresholding the HSV image using the lower and upper bounds
- Apply the mask to the cropped frame to extract the hand region

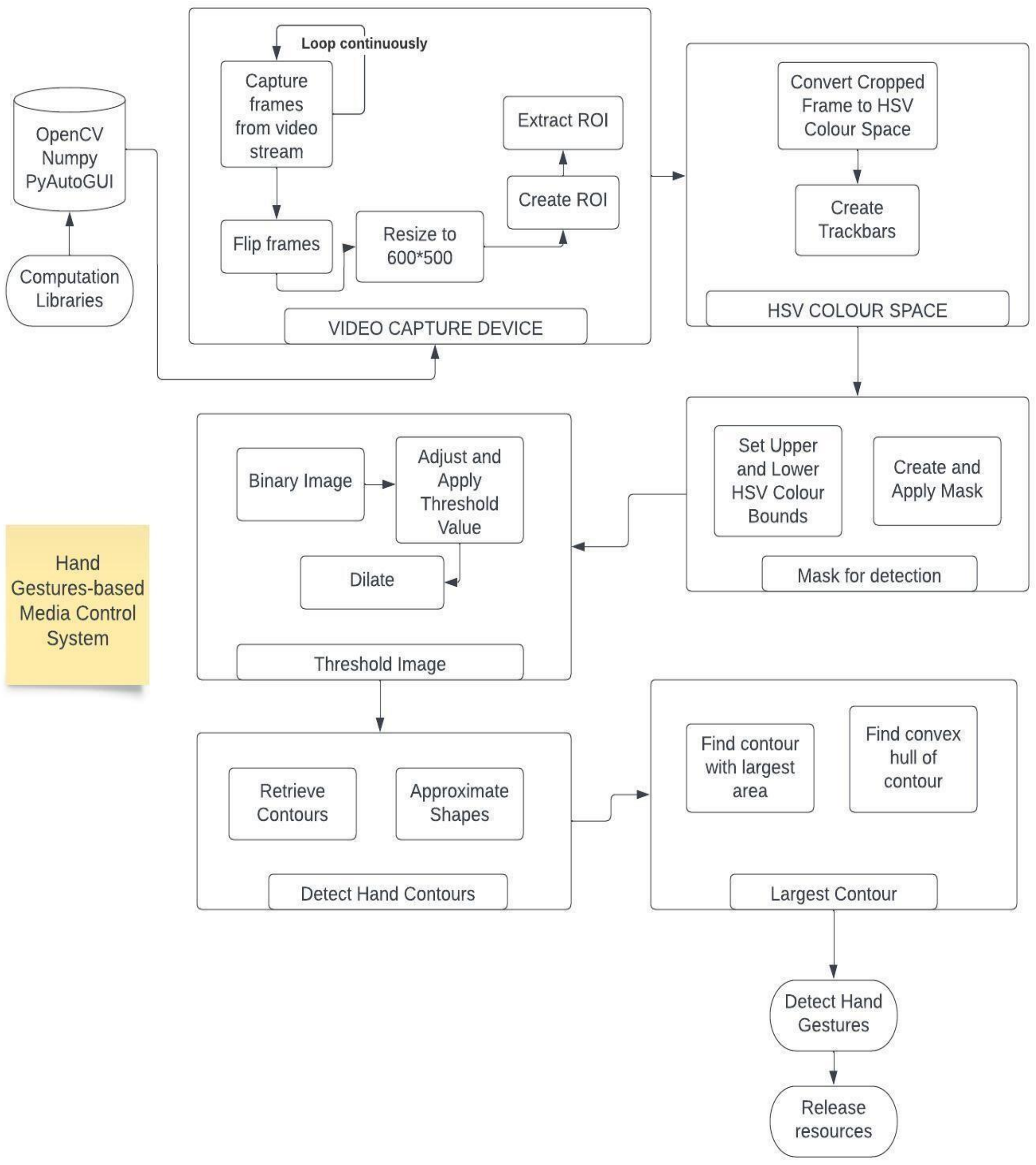
5. ***Threshold the image:***

- Invert the mask to obtain a binary image of the background
- Create a trackbar to adjust the threshold value for filtering out noise
- Apply a threshold to the binary image using the trackbar value to obtain a binary image of the hand

- Dilate the binary image to fill any gaps in the hand contour

6. ***Find the contours of the hand:***

- Use the findContours function to detect the hand contour
- Use the RETR_TREE and CHAIN_APPROX_SIMPLE flags to retrieve all the contours and approximate their shapes respectively



7. ***Find the largest contour:***

- Find the contour with the largest area in the list of contours
- Use the approxPolyDP function to simplify the contour by reducing the number of points
- Find the convex hull of the contour
- Draw the contour and convex hull on the original frame

8. ***Detect hand gestures:***

- Find the convexity defects of the contour using the convexityDefects function
- Calculate the angle between the fingers using the Cosine Rule
- If the angle is less than or equal to a threshold value (50 degrees), count it as a finger
- Display the number of fingers using the PyAutoGUI library to control the mouse and keyboard

9. ***Release the resources:***

- Release the video capture device and close all windows

IMPLEMENTATION

```
#Step -1
import cv2
import numpy as np
import math
import pyautogui as p
import time as t

#Read Camera
cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)
def nothing(x):
    pass
#window name
cv2.namedWindow("Color Adjustments",cv2.WINDOW_NORMAL)
cv2.resizeWindow("Color Adjustments", (300, 300))
cv2.createTrackbar("Thresh", "Color Adjustments", 0, 255, nothing)

#Color Detection Track

cv2.createTrackbar("Lower_H", "Color Adjustments", 0, 255,

nothing)

cv2.createTrackbar("Lower_S", "Color Adjustments", 0, 255, nothing)
cv2.createTrackbar("Lower_V", "Color Adjustments", 0, 255, nothing)
cv2.createTrackbar("Upper_H", "Color Adjustments", 255, 255, nothing)
cv2.createTrackbar("Upper_S", "Color Adjustments", 255, 255, nothing)
cv2.createTrackbar("Upper_V", "Color Adjustments", 255, 255, nothing)

while True:
    _,frame = cap.read()
    frame = cv2.flip(frame,2)
    frame = cv2.resize(frame,(600,500))
    # Get hand data from the rectangle sub window
    cv2.rectangle(frame, (0,1), (300,500), (255, 0, 0), 0)
    crop_image = frame[1:500, 0:300]

    #Step -2
    hsv = cv2.cvtColor(crop_image, cv2.COLOR_BGR2HSV)
    #detecting hand
    l_h = cv2.getTrackbarPos("Lower_H", "Color Adjustments")
    l_s = cv2.getTrackbarPos("Lower_S", "Color Adjustments")
```

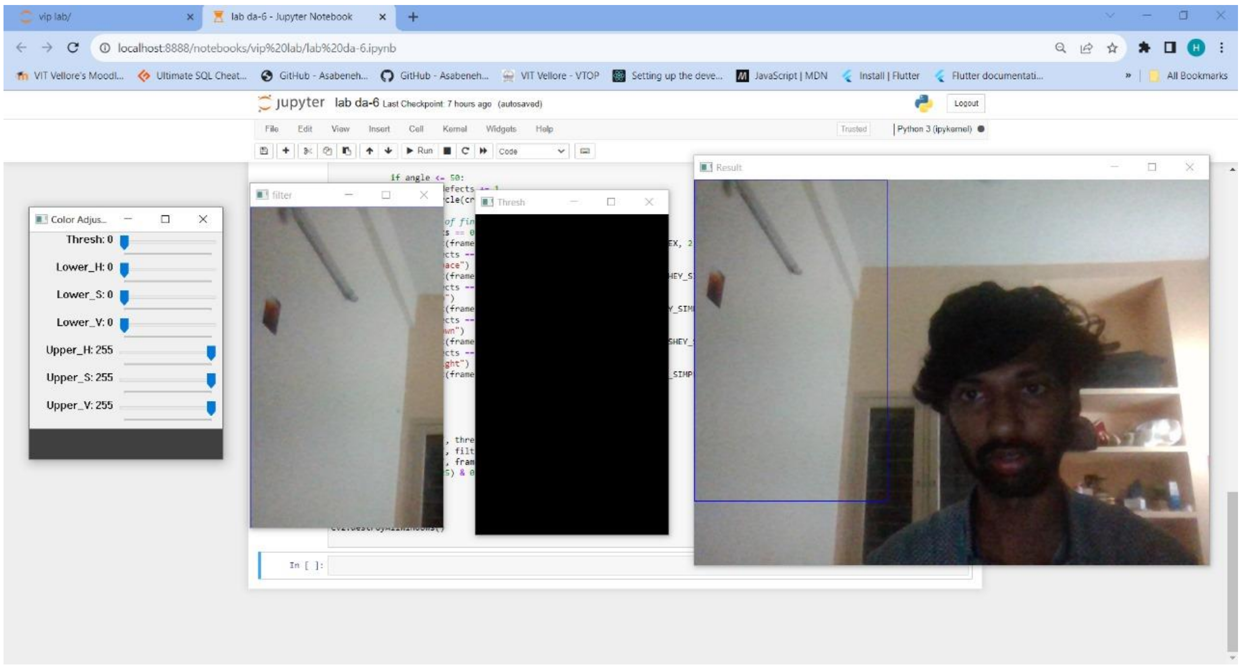
```
l_v = cv2.getTrackbarPos("Lower_V", "Color Adjustments")
u_h = cv2.getTrackbarPos("Upper_H", "Color Adjustments")
u_s = cv2.getTrackbarPos("Upper_S", "Color Adjustments")
u_v = cv2.getTrackbarPos("Upper_V", "Color Adjustments")

#Step -3
lower_bound = np.array([l_h, l_s, l_v])
upper_bound = np.array([u_h, u_s, u_v])

#Step - 4
#Creating Mask
mask = cv2.inRange(hsv, lower_bound, upper_bound)
#filter mask with image
filtr = cv2.bitwise_and(crop_image, crop_image, mask=mask)

#Step - 5
mask1  = cv2.bitwise_not(mask)
m_g = cv2.getTrackbarPos("Thresh", "Color Adjustments")

#getting trackbar value
ret,thresh = cv2.threshold(mask1,m_g,255,cv2.THRESH_BINARY)
dilata = cv2.dilate(thresh,(3,3),iterations = 6)
```

```
   #Step -6
   #findcontour(img,contour_retrival_mode,method)
   cnts,hier =
cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)

   try:
      #print("try")
       #Step -7
       # Find contour with maximum area
      cm = max(cnts, key=lambda x: cv2.contourArea(x))
      #print("C==",cnts)
      epsilon = 0.0005*cv2.arcLength(cm,True)
      data= cv2.approxPolyDP(cm,epsilon,True)

      hull = cv2.convexHull(cm)

      cv2.drawContours(crop_image, [cm], -1, (50, 50, 150), 2)
      cv2.drawContours(crop_image, [hull], -1, (0, 255, 0), 2)

      #Step - 8
      # Find convexity defects
      hull = cv2.convexHull(cm, returnPoints=False)
      defects = cv2.convexityDefects(cm, hull)
      count_defects = 0
      #print("Area==",cv2.contourArea(hull) - cv2.contourArea(cm))
      for i in range(defects.shape[0]):
         s,e,f,d = defects[i,0]

         start = tuple(cm[s][0])
         end = tuple(cm[e][0])
         far = tuple(cm[f][0])
         #Cosin Rule
         a = math.sqrt((end[0] - start[0]) * 2 + (end[1] - start[1]) * 2)
         b = math.sqrt((far[0] - start[0]) * 2 + (far[1] - start[1]) * 2)
         c = math.sqrt((end[0] - far[0]) * 2 + (end[1] - far[1]) * 2)
         angle = (math.acos((b * 2 + c * 2 - a ** 2) / (2 * b * c)) * 180) / 3.14
         #print(angle)
         # if angle <= 50 draw a circle at the far point
         if angle <= 50:
            count_defects += 1
            cv2.circle(crop_image,far,5,[255,255,255],-1)
```

```
        print("count==",count_defects)

        #Step - 9
        # Print number of fingersif
        count_defects == 0:

            cv2.putText(frame, " ", (50, 50), cv2.FONT_HERSHEY_SIMPLEX,
  2,(0,0,255),2)
        elif count_defects == 1:

            p.press("space")
            cv2.putText(frame, "Play/Pause", (50, 50),
cv2.FONT_HERSHEY_SIMPLEX, 2,(0,0,255), 2)
        elif count_defects == 2:
            p.press("up")

            cv2.putText(frame, "Volume UP", (5, 50),
cv2.FONT_HERSHEY_SIMPLEX, 2,(0,0,255), 2)
        elif count_defects == 3:
            p.press("down")

            cv2.putText(frame, "Volume Down", (50, 50),
cv2.FONT_HERSHEY_SIMPLEX, 2,(0,0,255), 2)
        elif count_defects == 4:
            p.press("right")

            cv2.putText(frame, "Forward", (50, 50),
cv2.FONT_HERSHEY_SIMPLEX, 2,(0,0,255), 2)
        else:
            pass

    except:
        pass #step
    -10
    cv2.imshow("Thresh", thresh)
    #cv2.imshow("mask==",mask)
    cv2.imshow("filter==",filtr)
    cv2.imshow("Result", frame)

    key = cv2.waitKey(25) &0xFF
    if key == 27:
        break
cap.release()
cv2.destroyAllWindows()
```

RESULTS

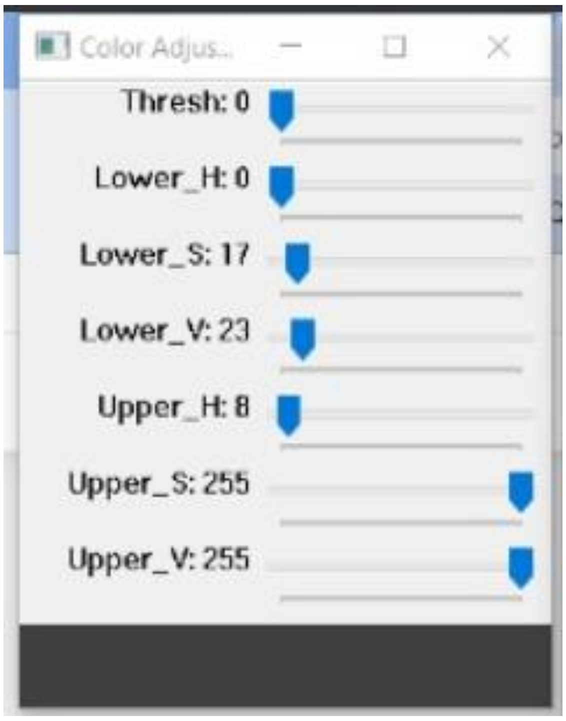
Figure 1 : Snapshot depicting entire media control system


vip lab/
lab da-6 - Jupyter Notebook
localhost:8888/notebooks/vip%20lab/lab%20da-6.ipynb
jupyter lab da-6 Last Checkpoint: 7 hours ago (autosaved)
Color Adjus...
filter
Thresh
Result

Color Adjus...

Thresh: 0
Lower_H: 0
Lower_S: 0
Lower_V: 0
Upper_H: 255
Upper_S: 255
Upper_V: 255

Figure 2 & 3 : Control Setting and hand detection screen


Color Adjus...

Thresh: 0
Lower_H: 0
Lower_S: 17
Lower_V: 23
Upper_H: 8
Upper_S: 255
Upper_V: 255

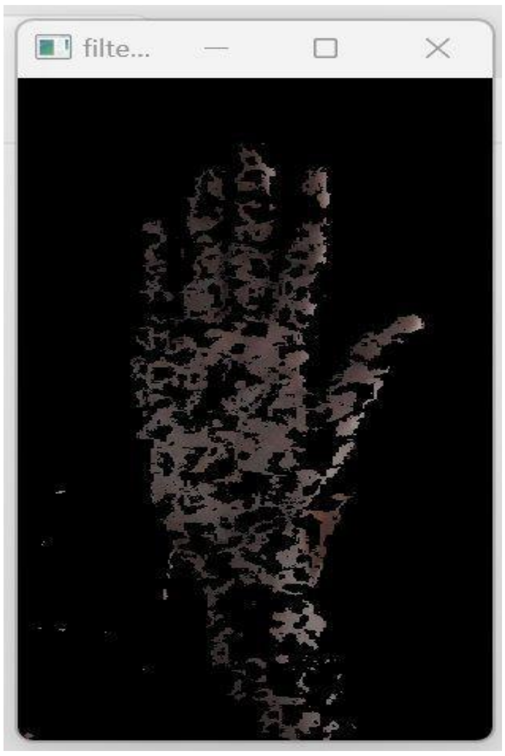


Figure 4 : Main screen recordedby video camera

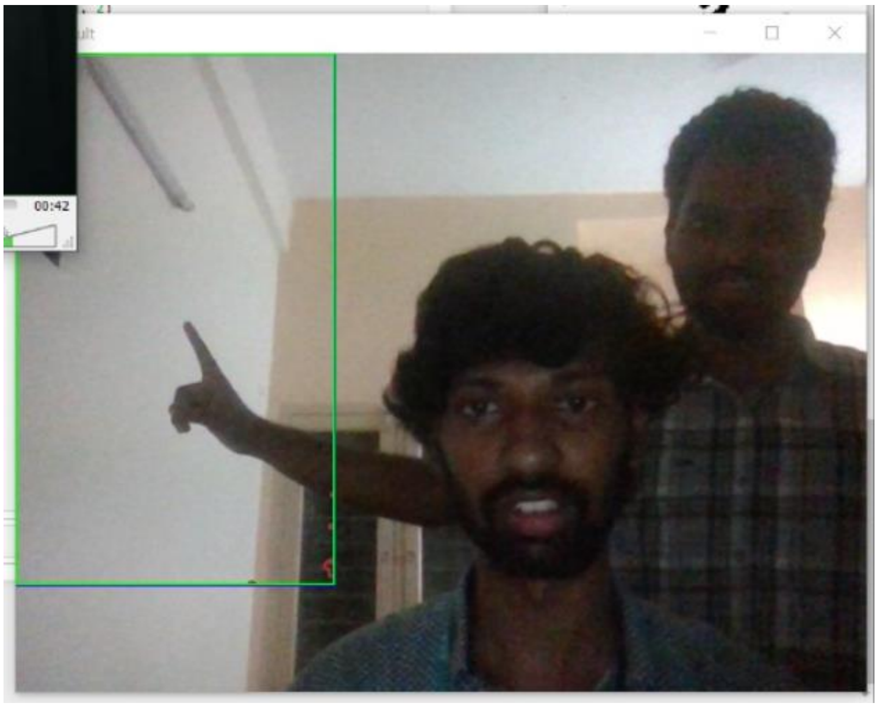
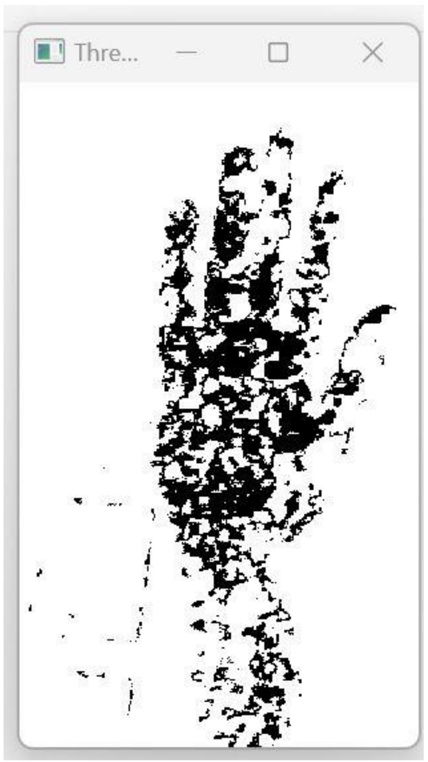


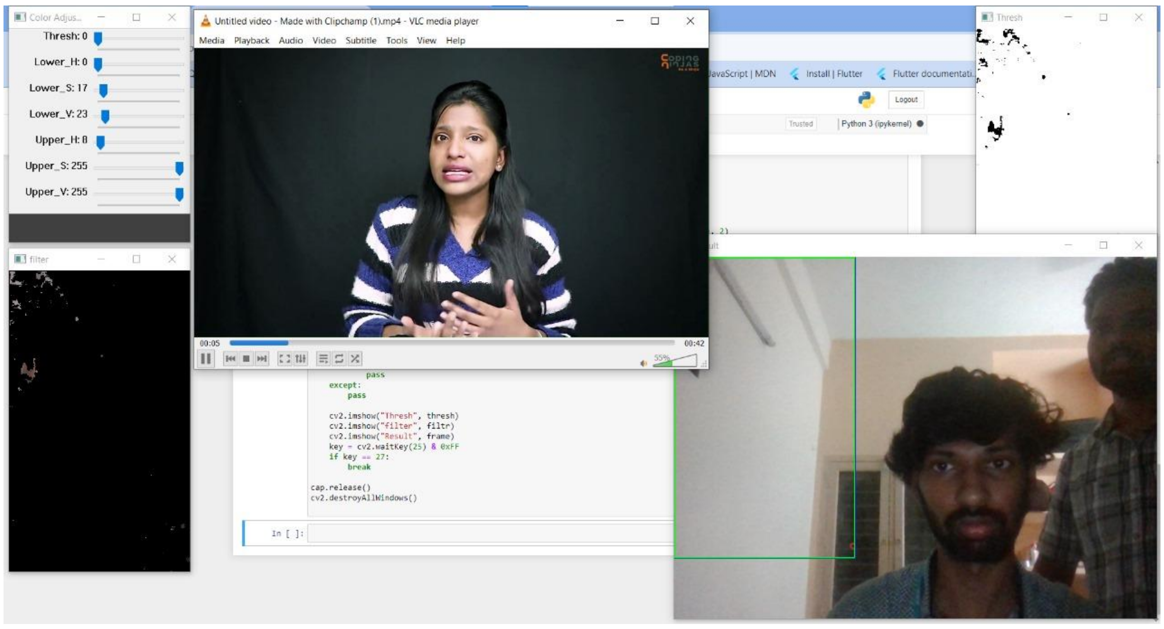
Figure 5 : Contours



All gestures

Gesture 1

Before

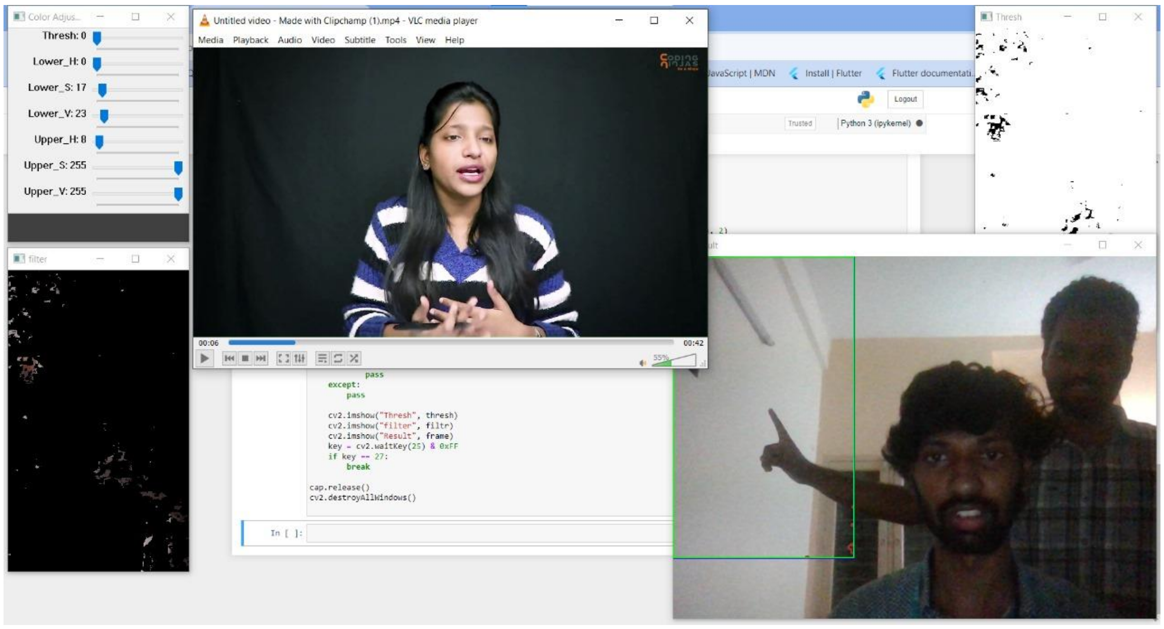


Thresh: 0
Lower_H: 0
Lower_S: 17
Lower_V: 23
Upper_H: 8
Upper_S: 255
Upper_V: 255

```
            pass
    except:
        pass

    cv2.imshow("Thresh", thresh)
    cv2.imshow("filter", filtr)
    cv2.imshow("Result", frame)
    key = cv2.waitKey(25) & 0xFF
    if key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

In [ ]:

After



Thresh: 0
Lower_H: 0
Lower_S: 17
Lower_V: 23
Upper_H: 8
Upper_S: 255
Upper_V: 255

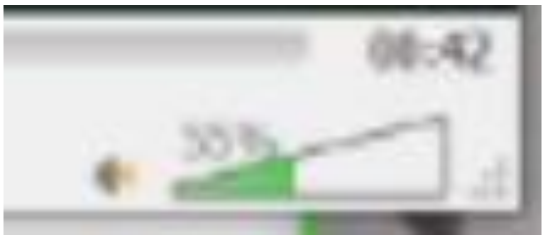
```
            pass
    except:
        pass

    cv2.imshow("Thresh", thresh)
    cv2.imshow("filter", filtr)
    cv2.imshow("Result", frame)
    key = cv2.waitKey(25) & 0xFF
    if key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

In [ ]:

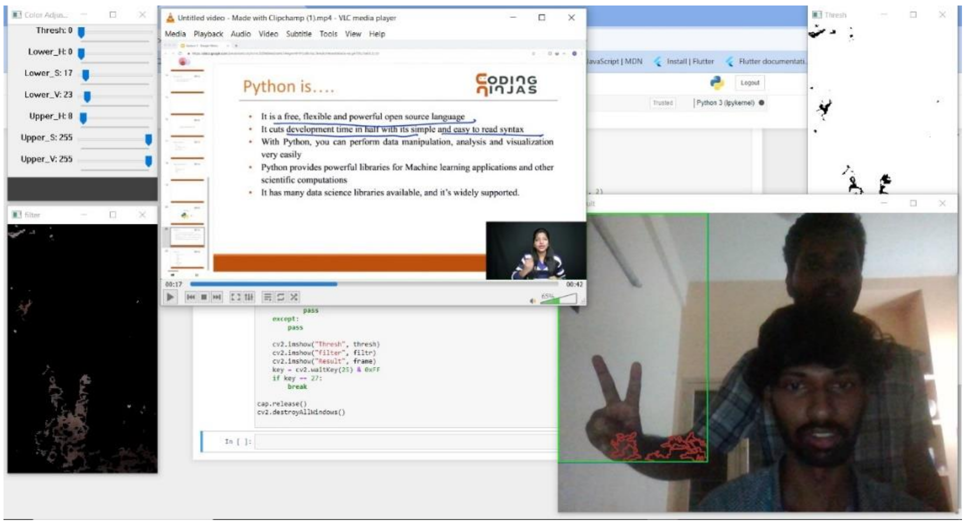
Video which was playing **is paused**

Gesture 2

Before (55 % volume)



After

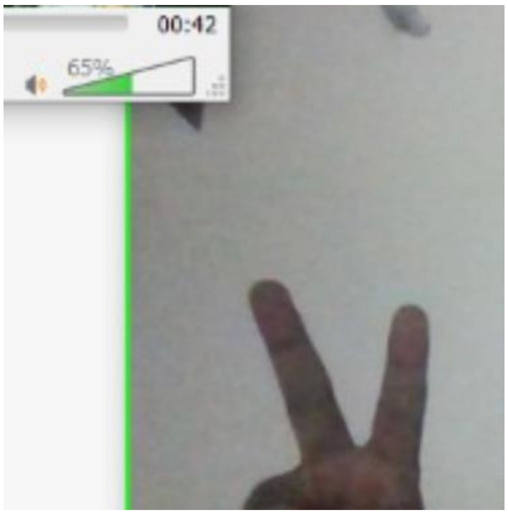

Color Adjus...
Untitled video - Made with Clipchamp (1).mp4 - VLC media player
Media Playback Audio Video Subtitle Tools View Help
filter
Thresh
00:17
00:42
65%

Thresh: 0
Lower_H: 0
Lower_S: 17
Lower_V: 23
Upper_H: 8
Upper_S: 255
Upper_V: 255

Python is....

- It is a free, flexible and powerful open source language
- It cuts development time in half with its simple and easy to read syntax
- With Python, you can perform data manipulation, analysis and visualization very easily
- Python provides powerful libraries for Machine learning applications and other scientific computations
- It has many data science libraries available, and it's widely supported.

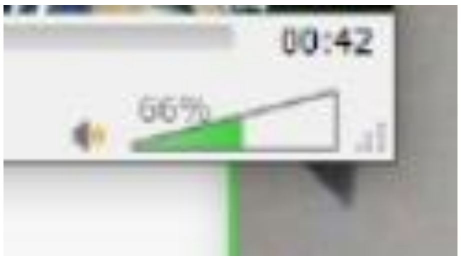
```
            pass
    except:
        pass

    cv2.imshow("Thresh", thresh)
    cv2.imshow("filter", filtr)
    cv2.imshow("Result", frame)
    key = cv2.waitKey(25) & 0xFF
    if key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```



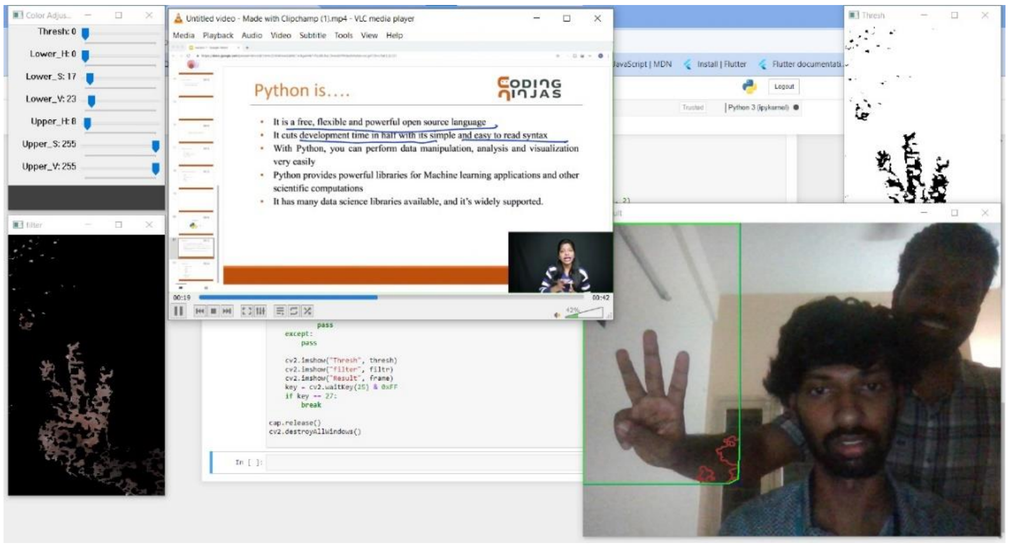
Video volume is increased to **65%**

Gesture 3

Before (Volume was 66%)



After



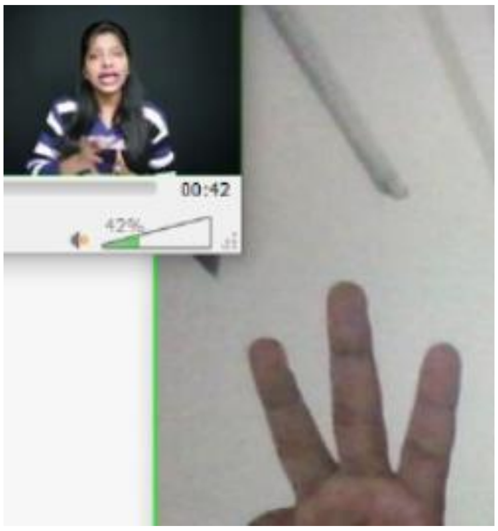
Python is....

COD I NG NINJAS

- It is a free, flexible and powerful open source language
- It cuts development time in half with its simple and easy to read syntax
- With Python, you can perform data manipulation, analysis and visualization very easily
- Python provides powerful libraries for Machine learning applications and other scientific computations
- It has many data science libraries available, and it's widely supported.

```
            pass
    except:
        pass

    cv2.imshow("Thresh", thresh)
    cv2.imshow("filter", filtr)
    cv2.imshow("Result", frame)
    key = cv2.waitKey(25) & 0xFF
    if key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

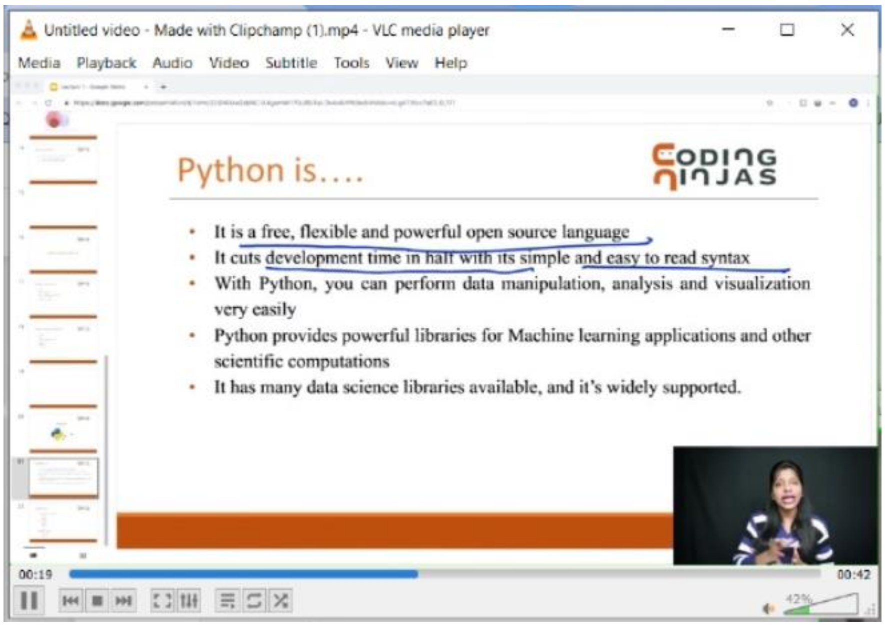
```
In [ ]:
```



Video Volume came down to **42%**

Gesture 4

Before (Video was playing at 00 : 19 seconds)

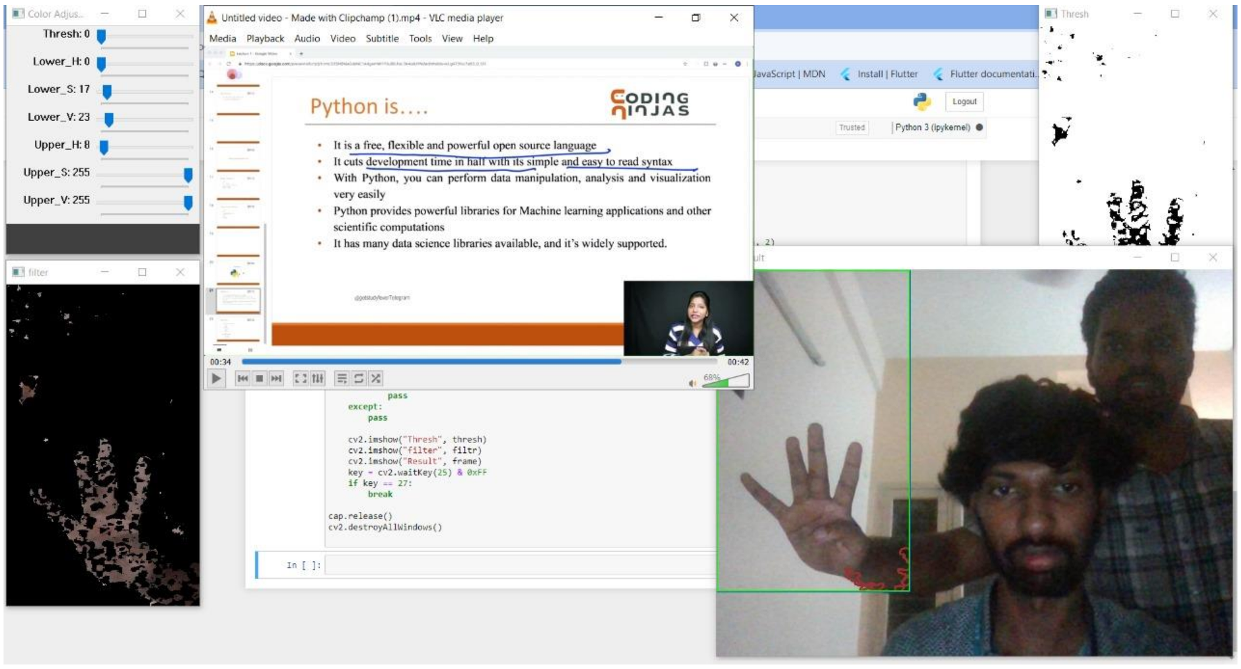

Untitled video - Made with Clipchamp (1).mp4 - VLC media player

Python is....

- It is a free, flexible and powerful open source language
- It cuts development time in half with its simple and easy to read syntax
- With Python, you can perform data manipulation, analysis and visualization very easily
- Python provides powerful libraries for Machine learning applications and other scientific computations
- It has many data science libraries available, and it's widely supported.

00:19 00:42

After


Color Adjus...

Thresh: 0
Lower_H: 0
Lower_S: 17
Lower_V: 23
Upper_H: 8
Upper_S: 255
Upper_V: 255

Untitled video - Made with Clipchamp (1).mp4 - VLC media player

Python is....

- It is a free, flexible and powerful open source language
- It cuts development time in half with its simple and easy to read syntax
- With Python, you can perform data manipulation, analysis and visualization very easily
- Python provides powerful libraries for Machine learning applications and other scientific computations
- It has many data science libraries available, and it's widely supported.

00:34 00:42

```
            pass
    except:
        pass

    cv2.imshow("Thresh", thresh)
    cv2.imshow("filter", filtr)
    cv2.imshow("Result", frame)
    key = cv2.waitKey(25) & 0xFF
    if key == 27:
        break

cap.release()
cv2.destroyAllWindows()
```

Video fast forwarded to 00:42 seconds

APPLICATIONS OF THE PROPOSED SYSTEM

A hand-gestures based media control system has the potential to improve the accessibility and usability of technology in a variety of applications, making it easier for users to interact with devices and software in a natural and intuitive way.

1. **Smart homes**: A hand-gestures based media control system could be used to control various smart devices in the home. For example, a user could raise their hand to turn on a light or wave their hand to adjust the temperature of a thermostat. This type of system could be particularly useful for individuals who have difficulty using traditional switches or touchscreens, such as those with mobility or vision impairments.
2. **Gaming**: Hand gestures could be used to control games, providing a more immersive gaming experience. For example, a user could use a hand gesture to mimic the motion of swinging a sword in a sword-fighting game or to perform a special move in a fighting game.
3. **Accessibility**: A hand-gestures based media control system could be particularly useful for individuals with disabilities, such as those who have limited mobility or difficulty speaking. The system could allow

them to control various devices and applications with simple hand gestures, making it easier for them to interact with technology.

4. **Presentations**: The system could be used to control presentations, allowing the presenter to navigate slides, start and stop videos, and perform other actions without the need for a remote or computer. This type of system could be particularly useful for presenters who need to move around a lot during their presentations, as it would allow them to control their slides without being tethered to a computer.
5. **Medical applications**: In a medical setting, a hand-gestures based media control system could be used to control various devices and equipment, such as surgical robots or medical imaging software. This could allow doctors and other medical professionals to interact with these devices more easily and efficiently, potentially leading to better patient outcomes.
6. **Virtual and augmented reality**: Hand gestures could be used to control virtual and augmented reality environments, allowing for more intuitive and natural interaction with these technologies. For example, a user could use hand gestures to move objects in a virtual environment or to select options in an augmented reality display.
7. **Automotive industry**: In the automotive industry, a hand-gestures based media control system could be used to control various features of a car, such as the infotainment system, climate control, and navigation. This could allow drivers to interact with these features without taking their hands off the steering wheel or their eyes off the road, increasing safety. For example, a user could use a hand gesture to adjust the volume of the car's sound system or to answer a phone call hands-free.
8. **Education**: In an educational setting, a hand-gestures based media control system could be used to control presentations, display content on interactive whiteboards, or to navigate educational software. This could make it easier for teachers to interact with technology during lessons and for students to participate in activities without needing to touch shared devices or equipment. For example, a student could use hand gestures to control a robot during a programming class or to manipulate virtual objects during a science experiment.
9. **Industrial settings**: In industrial settings, a hand-gestures based media control system could be used to control machinery, monitor production lines, or access technical data. This could allow workers to interact with machines and systems more efficiently and safely, reducing the risk of injury or error. For example, a worker could use hand gestures to start or stop a machine, to adjust its settings, or to access information about its performance.

10. **Retail industry**: In the retail industry, a hand-gestures based media control system could be used to control various features of a store, such as interactive displays, product demos, and checkout systems. This could allow customers to interact with products and services in a more engaging and immersive way, potentially increasing sales. For example, a customer could use hand gestures to browse through a digital catalog, to see a virtual product demo, or to pay for their purchases without needing to touch a payment terminal.

NOVELTY IN OUR IDEA

Hand gesture recognition technology is still relatively new, and as such, there is still room for innovation and improvement in this area. Advances in computer vision and machine learning algorithms have made it possible to develop more accurate and reliable hand gesture recognition systems, which can lead to more efficient and effective control of media devices.

Our proposed media-controller system effectively segregates the screen for better detection of hand movements as opposed to many already-existing prototypes giving rise to unwanted hand-gestures detection.

We have also incorporated colour adjustment settings and HSV settings for better detection of hands by the video camera or rgb sensor.

Inside the while loop, it reads the frame from the camera, crops it to a smaller region of interest, and applies a series of image processing operations to it.

This precise and quick detection overall increases the efficiency and reduces the latency.

REFERENCES

[1] G. Krastev and I. Ralev, "Gesture Based System for User Interface Control," 2021 5th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, IEEE 2021.

[2] Monisha Sampath, Priyadarshini Velraj, Vaishnavii Raghavendran, and M Sumithra, “Controlling media player using hand gestures with VLC media player,” World Journal of Advanced Research and Reviews, vol. 14, no. 3, pp. 466–472, Jun. 2022.

[3] Bilvika, K & K, Sneha & M, Sahana & Patil, Tejaswini. (2021). “Face and Hand Gesture Recognition System for Controlling VLC Media Player”. International Journal of Scientific Research in Science and Technology.

[4] Sakshi Shinde, Sarthak Mushrif, Aditya Pardeshi, Dhairyasheel Jagtap, Prof. Vandana Rupnar. “Gesture-based Media Player Controller”. International Journal of Research Publication and Reviews, May 2022.

[5] P. Maloo, “Controlling Media Player with Hand Gestures,” International Journal of Engineering Research & Technology, vol. 11, no. 11, Nov. 2022, doi: 10.17577/IJERTV11IS110088.

[6] G. D. Nagalapuram, R. S, V. D, D. D and D. J. Nazareth, "Controlling Media Player with Hand Gestures using Convolutional Neural Network," 2021 IEEE Mysore Sub Section International Conference (MysuruCon), Hassan, India, 2021, pp. 79-86, doi: 10.1109/MysuruCon52639.2021.9641567.

[7] K. Venkatasalam, Sourav, V. Yashwanth Venkata, S. Chowdary, and R. Chandu, “AUDIO PLAYER CONTROLLING BASED ON HAND GESTURE TECHNIQUE,” 2604. International Research Journal of Modernization in Engineering Technology and Science, June, 2022.