Hand gesture based media player

Akshith Reddy Kota

December 9, 2022

Abstract

We all consume a lot of video content on a daily basis in the form of Youtube videos, Instagram, Facebook, etc. It is not uncommon to see people finding it difficult to operate a media player playing a particular video while multitasking. In this day and age, it is required by many people to multitask in order to get things done. I personally find time to watch videos and entertain myself while I am having my lunch and dinner. And, I find it quite a hassle to operate the keyboard of my laptop in order to play/pause the video, increase or decrease its volume or for that matter fast forward or rewind the video. There might be a chance of spilling my food on the laptop. Furthermore, in this way, people might find it difficult to operate a keyboard of a laptop while they are doing another task as it may hamper their ability to successfully perform the task at hand. In this project, I have developed a hand gesture based media player that can take in the gestures of our hands and depending on those gestures, it can control the video playing in a particular media player. I have implemented the following five functionalities in this media player- palm with no open fingers will play or pause the video, showing three fingers will increase the volume of the video, four fingers will decrease the volume of the video and finally five fingers to fast forward the video. For that matter, we can convert any type of gesture to any type of command by using this hand gesture based media player.

Keywords: Hand gesture, Gesture Recognition, Human Computer Interaction, Computer Vision, Media Player

1 Introduction

The importance of gesture recognition comes in the fact that this type of recognition will allow people to communicate with the machines in a more robust way without needing to have any prerequisite knowledge on how to operate a machine[1]. Employing a gesture based system in a framework will allow its users to have more accessibility while improving the user experience[2].

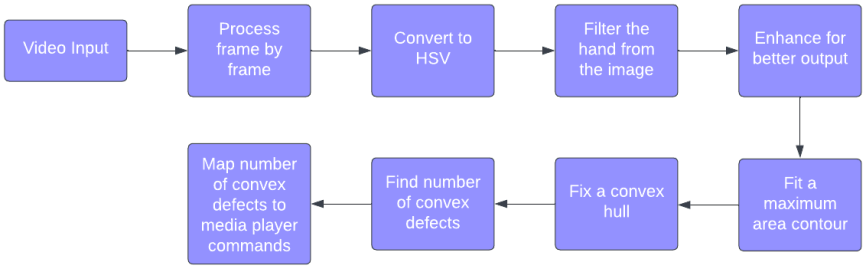

Figure 1: Architecture

Although there are many type of gesture recognition available, hand gesture recognition is concise, easy to understand and implement on a large scale. However, one disadvantage of hand recognition is that the recognition itself depends on the amount of light present in the environment where the recognition is taking place and the distance of the hand from the camera[3]. So the recognition system must give the users the ability to tune the hyperparameters of the system in order for them to dynamically change the hsv values and improve the recognition of their hands even in a dim lit environment [4].

This kind of gesture based recognition needs a one to one mapping between the gesture and the command that particular gesture is intended to perform. If the mapping is many to one, then the program might not work as expected because for two or more gestures, there is a single command if the function mapping from gestures to commands is not one to one[5].

One can also enhance and improve the hand recognition using various image processing techniques like dilation[6] in order to remove the noise from the input image, etc. This kind of hand gesture recognition can also be used to operate music in a media player. This paper aims to highlight the aspect of hand gesture recognition for operating videos playing in a particular media player. In figure 1, you can see the architecture of the proposed hand gesture based media player.

2 Background Study

In this section, let us discuss the steps involved in implementing the hand gesture based media player.

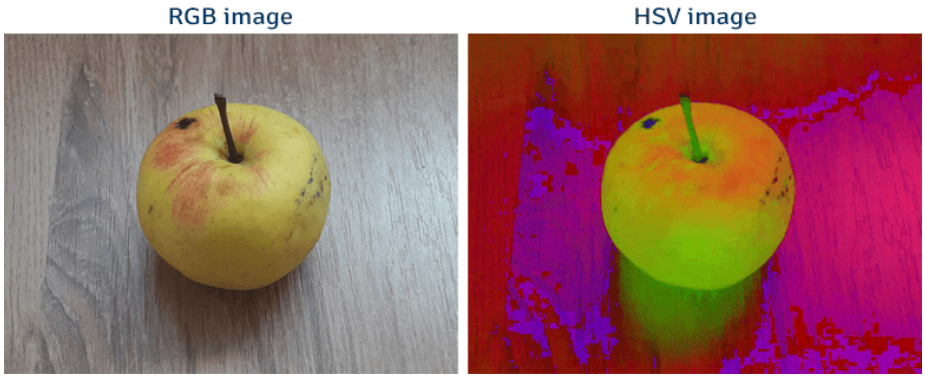

Figure 2: RGB image to HSV image
Image credits:
https://lindevs.com/convert-image-from-rgb-to-hsv-color-space-using-opencv

2.1 Import the relevant libraries and start capturing the camera

OpenCV library was used to capture the video input from the webcam of the laptop[7]. Furthermore, Pyautogui library was used to automatically press the key on the keyboard based on a particular condition rather than me manually pressing the key which goes against the aim of the project[8]. Additionally, OpenCV is also used to perform all the necessary image processing like inversion, masking, dilation, etc.

2.2 Access the video frame by frame and convert each frame into hsv

In the next step, the video was accessed frame by frame and each of this frame is first converted to an HSV image where the user was given the flexibility to adjust the HSV values in order to properly detect the hand. This step was performed in order to extract the information of light luminance from the image and ignore all the color information.

2.3 Construct a trackbar so that one can adjust the hsv values to detect the hand properly

As mentioned in the introduction, the performamce of the hand gesture recognition will depend in the luminance of the environment in which the recognition is taking place and the distance of the hand from the camera. So a trackbar was constructed to give the user the ability to dynamically choose the HSV values.

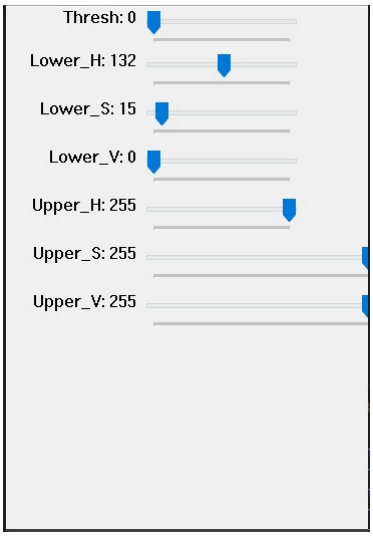

Figure 3: Trackbar to adjust the HSV values

On top of this, the trackbar was developed in such a way that the user can select the upper and lower bounds of the HSV values. Before starting the hand recognition, the user needs to adjust these values in order to start recognizing the hand. One downside is that if the hand is not properly recognized, then the output will not be correct and the program will perform unintended actions.

2.4 Filter the image based on the values set in the trackbar

Filter the image based on the values set in the trackbar. This essentially creates a mask to detect our hand. Only the pixels in the frame that are in the range between the lower bound of hsv values and the upper bound of the hsv values will get filtered. This is done to extract only our hand based on the color of our skin and ignore all the background information.

2.5 To achieve a better output, invert the pixel values and enhance the result

The extracted hand is then inverted to make it white while the color of the background is black. This was done to get much better output and enhance the result. On top of this, dilation was performed to eliminate the noise that is present in the image. So, essentially, the final output is a black and white image with our hand (white color) in the foreground and a black background.

Figure 4: Detected hand from a particular frame

2.6 Find the contours for the specified colour object

A contour is basically an outline representing the shape of an object, the object being our hand in this case.There are many contours that can be extracted for the recognized hand. So, we basically give the color range in the trackbar, then all the pixels in that range will get extracted and enhanced, and then, contours are fitted to those extracted pixels.

2.7 Find the contour with the maximum area and fit a convex hull

As there are many contours, we find the one with the maximum area and fit what is called a convex hull. To understand this term, break it into two parts-convex and hull. A convex shape is a shape where all the interior angles are less than 180 degrees and a hull means the exterior or a shape of an object. So a convex hull is a convex shape fitted to our contour. You can see the convex hull as the green colour shape surrounding the hand in the figure below.

2.8 Find the number of convex defects

A defect is basically an abrupt change in the shape of our convex hull. We use this count to operate the video, more specifically pause it, play it, increase its

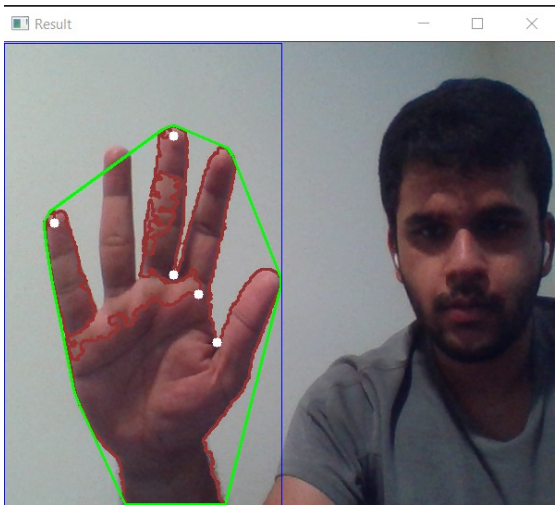

Figure 5: Convex hull fitted to the maximum area contour

volume, decrease its volume and fast forward it. The white dots in the above image are the convex defects and we will map the number of convex defects to a particular command for the media player. This way we can use just the number of convex defects in the convex hull fitted to the maximum area contour of our hand to operate the video in any media player.

3 Technical Implementation

In this section, let us see the technical implementation of the hand gesture based media player.

3.1 Play/Pause the video

If the count of convex defects is 1, then the space bar on the keyboard is pressed by pyautogui thereby playing or passing the video. If we open our palm without any gaps between our fingers as shown in the picture below, then the video will be paused if is was playing or vice versa.

3.2 Increase the volume of the video

If the number of convex defects is 2, then the 'up' key on the keyboard is pressed through pyautogui library. For this, we need to open three fingers so that the number of convex defects in the convex hull will be 2 as shown in the figure.

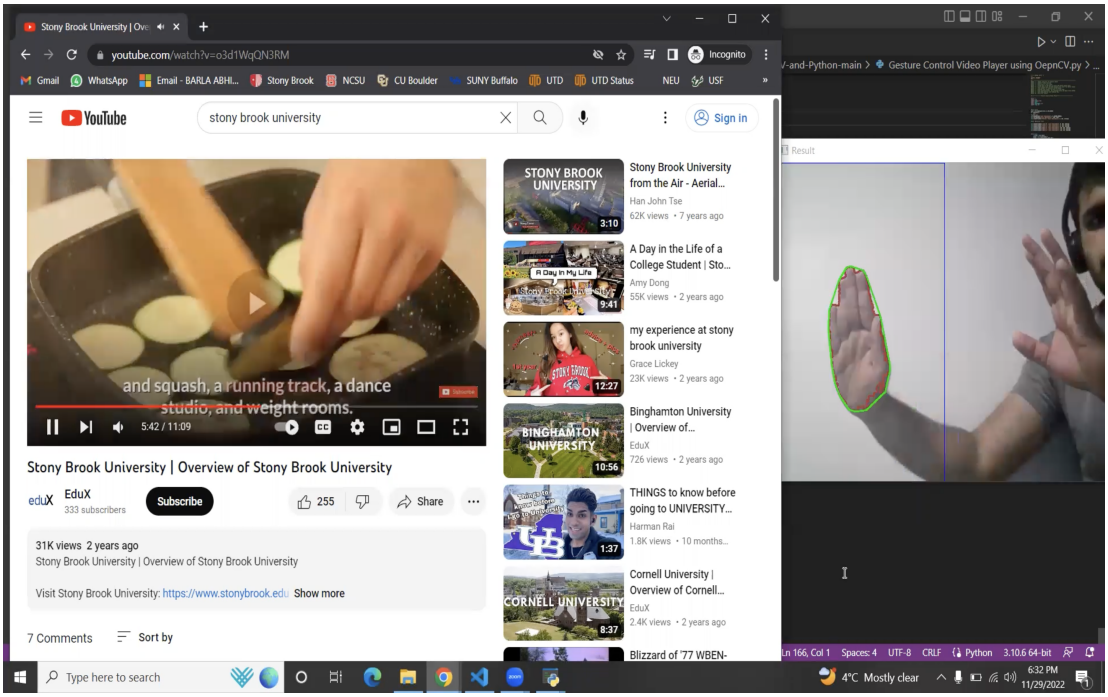

Figure 6: Playing/Pausing the video

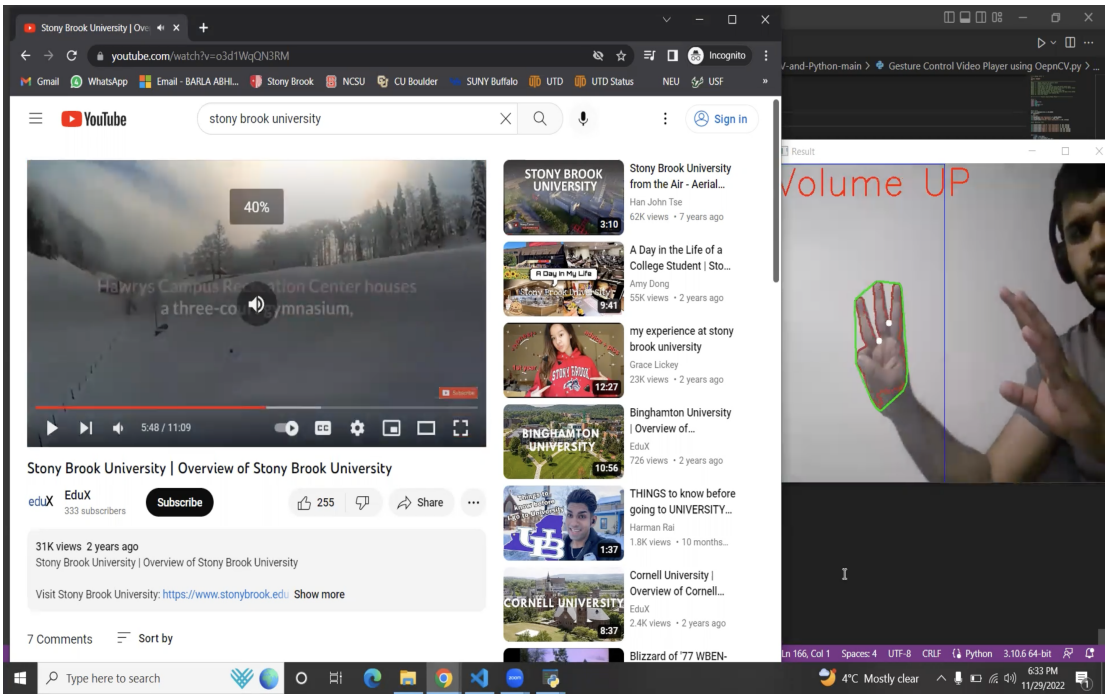

Figure 7: Increasing the volume of the video

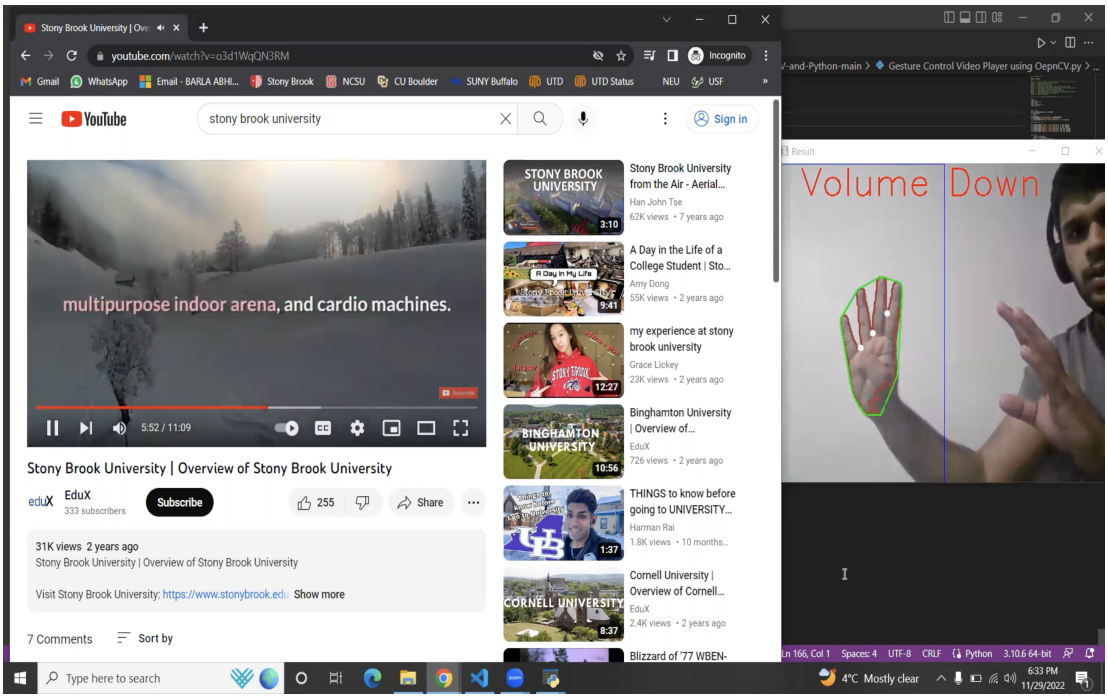

Figure 8: Decreasing the volume of the video

3.3 Decrease the volume of the video

If the number of convex defects is 3, then the 'down' key on the keyboard is pressed through pyautogui library. For this, we need to open four fingers so that the number of convex defects in the convex hull will be 3 as shown in the figure.

3.4 Fast forward the video

If the number of convex defects is 4, then the 'right' key on the keyboard is pressed through pyautogui library. For this, we need to open five fingers so that the number of convex defects in the convex hull will be 4 as shown in the figure.

4 Validation

The efficiency of this hand gesture based media player depends on its ability to accurately find the number of convex defects in the convex hull. So user testing has been done on the accuracy of the convex defects detection. A total of 15 users in one group and 25 users in the second group have controlled a particular

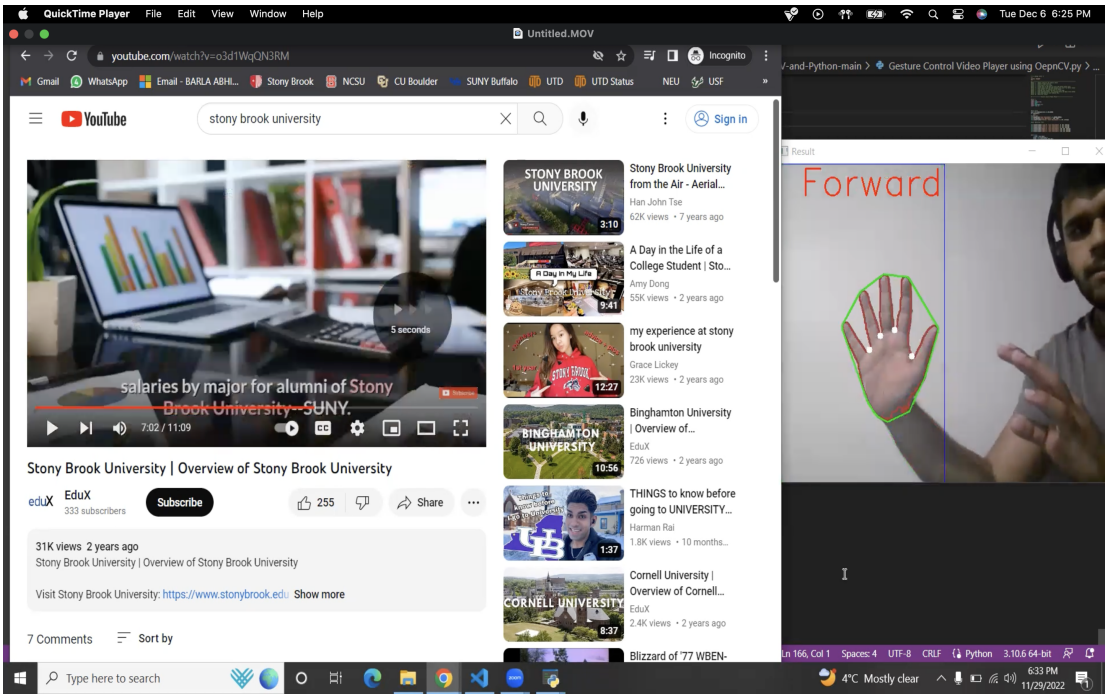

Figure 9: Fast forwarding the video

youtube video based solely on hand gestures. Each user was asked to perform all the hand gestures ,that being the one to pause/play the video, one to increase the volume, one to decrease the volume and the last gesture to fast forward the video. The true labels of the gestures and the ones that were predicted by the program using the number of convex defects were noted down and were compared. In the case of the first user group, the accuracy was 96.723 percent and for the second group, the accuracy was 95.895 percent. As one can see, the accuracy remained almost the same even as the number of users were increased in the second study. Therefore, one can say that the program is working as intended and is recognizing the hand gestures with a high accuracy. One point to note down here is that in the second study, for one user, the environment was dimly lit and hence the accuracy for the one user was relatively low. This confirms the assumption that the hand recognition depends on the luminance in the room.

5 Discussion and Future Work

The mappings from gestures to commands for the media player can be changed as the input is the number of convex defects and we can assign any command to

a particular number of convex defects. Furthermore, one can also use gestures of two hands instead of one by masking out the images of two hands, fitting a maximum area contour, convex hull and finally find the number of convex defects. In this situation, we will have more options for the commands as we will be able to map a maximum of 10 convex defects to commands. This way we can operate the media player in a more sophisticated way. .

6 Conclusion

Conventional method to operate a video using a keyboard can be quite a hassle when people multitask. Using the hand gesture based media player, one can overcome this hassle and operate the media player just by using hand gestures. This technology can also be used in smartphones and the above mentioned problem can also occur when people are using their smartphones. Also, the mentioned architecture in the report supports scalability as almost every step can use parallel computing and speed up the process of hand recognition. Furthermore, using a 60 or 120 fps webcam will further increase the accuracy of the hand recognition which in turn leads to a better control on the media player using the hand gestures. Thus, by incorporating parallel computing and sophisticated camera hardware, one can further the accuracy of the aforementioned architecture and create much better hand gesture based media players. This technology can be used on any type of media player, be it audio only or video only or both.

7 References

1)Yasen, M., Jusoh, S. (2019). A systematic review on hand gesture recognition techniques, challenges and applications. PeerJ Computer Science, 5, e218.
2)Oudah, M., Al-Naji, A., Chahl, J. (2020). Hand gesture recognition based on computer vision: a review of techniques. journal of Imaging, 6(8), 73.
3)Mo, T., Sun, P. (2020). Research on key issues of gesture recognition for artificial intelligence. Soft Computing, 24(8), 5795-5803.
4)Harini, V., Prahelika, V., Sneka, I., Adlene Ebenezer, P. (2018, November). Hand gesture recognition using OpenCv and Python. In International Conference On Computational Vision and Bio Inspired Computing (pp. 1711-1719). Springer, Cham.
5)Bashir, A., Malik, F., Haider, F., Ehatisham-ul-Haq, M., Raheel, A., Arsalan, A. (2020, January). A smart sensor-based gesture recognition system for media player control. In 2020 3rd International Conference on Comput-

ing, Mathematics and Engineering Technologies (iCoMET) (pp. 1-6). IEEE. 6)Saman, O., Stanciu, L. (2019, October). Image Processing Algorithm for Appearance-Based Gesture Recognition. In 2019 23rd International Conference on System Theory, Control and Computing (ICSTCC) (pp. 681-684). IEEE. 7)Rupnar, V. GESTURE BASED MEDIA PLAYER CONTROLLER. Journal homepage: www. ijrpr. com ISSN, 2582, 7421. 8)Balamurugan, P., Santhosh, J., Arulkumaran, G. (2019). Hand motion based mouse cursor control using image processing. Journal of Critical Reviews, 7(4), 2020.