Media Player Controlling Using Hand Gestures

Marasakatla Devendar, Rasamadugu Aravind, Boya Bhaskar Raju Challa Rama Krishna

Abstract—As computing becomes more and more ubiquitous in society, we must understand how to interact with computers through various means - one of which is hand gestures. Human-computer interaction is necessary for many applications, and as such, many studies have been conducted to classify hand gestures so that they might be better understood and utilized in computer applications. Automated vehicle movement systems have benefited greatly from recognizing human hand gestures, and this paper details a Python-based approach to detecting and recognizing them. This is the specific process flow of image processing includes the steps of background subtraction, hand ROI segmentation, contour detection, and finger detection. - all of which is done by utilizing a transfer learning CNN-trained model. The image's hand region is first separated from the rest of the background using thresholding, and then the contours of the hand are recognized and gestures are detected to edit the image. This system has two main advantages: It is extremely easy to transplant these modularized add-on layers to other applications because each step is isolated from the others and it doesn't need a lot of training data to achieve high accuracy. The hand gesture detection/recognition method proposed in this paper combines a deep learning CNN approach with an enhancement technique to achieve high performance. By using OpenCV, which provides a library collection of functions for various image processing techniques, the input images can be processed and corresponding keystrokes generated.

Index Terms— CNN classifier, Face Recognition, Face Detection, Threshold, Segmentation, Region of interest, OpenCV.

1 Introduction

Gesture recognition is a natural and intelligent way to interact with your computer. It can be used in various applications, such as medical rehabilitation and consumer electronics control. There are various sensing techniques used to identify hand gestures. The pc can be configured to recognize simple gestures to perform pattern recognition. Gesture recognition plays a key role in operating a computer in a distant mode. The computer considers the gestures of the human hand and recognizes them as part of its operation. Two types of gestures are used in this process: dynamic and static. The former allows the gestures to remain stationary while the machine is being operated, while the latter changes its position during the operation.

Dynamic gestures are more important than static gestures. First, a camera attached to the machine records human gestures. The foreground of the gesture is captured after the background of all detected gestures is removed. After segmentation of the provided images, the handheld ROI images are removed. Thresholding is a segmentation technique employed in this case and has a straightforward preparation algorithm. We will extract the outlines after thresholding. We will then discover the convex hull and the angle between the fingers. We will count fingers based on these computations. The gesture-based machine operating system is used in numerous consumer electronics a car without human contact. Human gestures can be classified as static or dynamic, along with online or offline. Offline gestures can only be used to control the machine's icons; they cannot be used to change the system's or menu's order of things. The machine's icons are controlled by internet motions.

The following is how the paper is set up: Deals with the traditional methods of hand gesture recognition before proposing CNN classification-based hand gesture recognition and talking about the simulation outcomes of the suggested hand gesture recognition system. The study concludes with one application that makes use of the hand gesture recognition idea.

One of the difficult tasks in designing a hand gesture recognition system in this study involves two main problems. The recognition of hands is the first step. Real-time video capture with a camera enables the detection of the user's hands. The issue would be conflicting brightness, noise, poor image quality, and Poor contrast. Hands are recognized to identify gestures in the video. The process at this point includes segmentation, edge detection, and background subtraction.

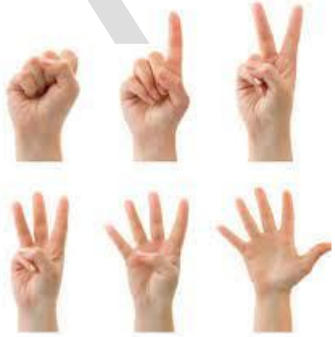


Fig. 1 Different hand gesture postures

2 Citations

1. Zuocai Wang et al. (2018) proposed a hand gesture recognition system based on particle filtering and tested it using hand gesture images with a constant background. The authors found that this filtering method was effective in accurately classifying the images.
2. Suguna and Neetu (2017) extract morphological features from hand gesture images, which is a good way to classify them. The k-means clustering algorithm is a great tool for training and classifying these images.
   Marium et al. (2017) proposed hand gesture recognition. system based on the convexity algorithm. This

filtering method works great on images of hand gestures with the same background.

3. The authors created a Markov model to study foreground fingers in images of hand gestures. This model was then used to train and test a binary classification approach.
4. Rahman and Afrin (2013) attempted to classify hand gesture images using the SVM (Support Vector Machine) classification method, but the error rate was too high to be practical, making it unusable for fast-moving images with lots of background or foreground objects. Rao et al. (2009) used a hidden Markov model to create a hand gesture recognition system which avoided this issue.

3 Proposed Methodology

In this study, the CNN classification approach was used to identify and detect human hand motions. The steps in this procedure include Region-of-interest segmentation of a hand using a mask image, finger segmentation, normalization of segmented finger images, and finger recognition using a CNN classifier. The proposed hand gesture recognition system algorithm is as follows.

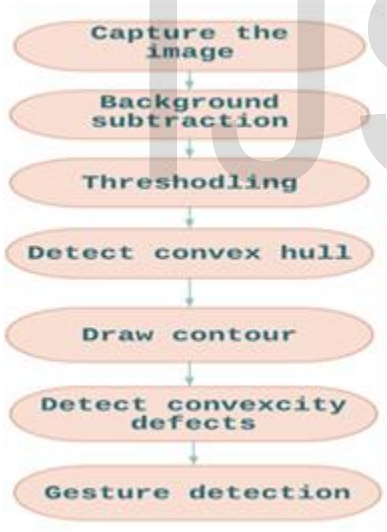


Fig. 2 Process of hand gesture recognition

The system was built using Python, the Python OS module, and the OpenCV package. In this case, we also used a pre-trained CNN-based transfer learning model for hand gesture recognition. Transfer learning is a deep learning technique that allows for a model created for one task to be used as the foundation for a model for another.

A deep learning convolutional neural network is commonly used for tasks such as hand detection and recognition. were employed to test and train the model. The models were created using Google's Kera and TensorFlow programs. TensorFlow operates on the backend while Kera is used on the front end to training a model in Python.

3.1 Image Capturing

Simply using my bare hands sans gloves, I acquired the RGB image in this phase using a camera (frame by frame).

3.2 Pre-Processing

In this step, only significant parts of the frames are removed from the video stream to reduce computational time, known as the Region of Interest (ROI).

Image processing prefers to convert colour images to grayscale for better processing. of interest is converted to grayscale and then restore the photos to their original colour space once processing is finished. The objects with a high frequency but not the goal is then reduced by blurring them with a Gaussian filter (ROI).

3.3 Hand Region Segmentation

We were able to The region of interest (ROI) step reduces computation time by removing only important parts of frames from the video stream. Image processing prefers to convert colour images to grayscale for better processing., so we convert the region of interest to grayscale. Then, we blur objects with a Gaussian filter (ROI) that have a high frequency but are not the goal. Finally, we restore the photos to their original colour space.

1. Edge detection.
2. The reason the hand has a different colour in the image is that its RGB values are different from the background's RGB values.
3. The background subtraction

In this article, we used a background subtraction technique to isolate the hand from the background. This method focuses on determining the background by utilizing the Running Average Principle. The background calculation is based on a system that focuses on a specific scene for at least 30 frames and calculates the average over the current frame and all previous frames using the following equation: dst(x,y)=(1−a).dst(x,y)+a.src(x,y), The moving average over frames is determined by the weights of the original image. (alpha) and the number of channels in the original image (x, y). The final image will have the same number of channels as the original image but will be an 8-bit or 32-bit floating point. After defining the background, place the hand in front of the camera and use a moving average to calculate the absolute difference between the computed background and the current frame with the hand as the foreground object. So, find the background and calculate the difference, referred to as the subtraction background. The image is then thresholded in the next step. Subtract the background so only the hand is white and the rest of the image is black. To achieve high accuracy, the thresholding process must be completed before the contour retrieval process. The threshold principle can be expressed mathematically as, where f(x) is the pixel's intensity.

This principle helps us understand how pixels with different intensities are thresholded, and how this affects the overall image. Motion-Detection is a process that refers to all of the processes described above. Figure 3 below depicts the results of the Hand region segmentation process. To remove any small noise regions, a series of morphological processes like erosions and dilations can be used.

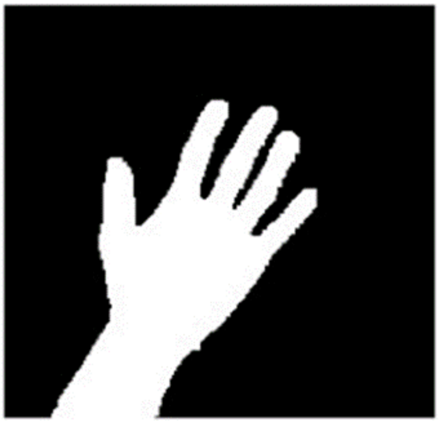


Fig. 3 hand region segmentation

3.4 Contour - Extraction

In image analysis, an outline is defined as the boundary or outline of an object. In other words, contours are curves that connect points with equal colour values and are especially important for geometry analysis, object detection and recognition. Outlines can be used to define the size, shape, and orientation of objects in an image.

3.5 Extraction and Recognition of Characteristics

Next, we need to determine how many fingers are being used in the hand gesture. You can do this by checking the number of points in the convex hull that encloses the arm region. As shown in Figure 4, the convex hull is a group of points enclosing the arm region.

The extreme points (top, bottom, left, and right) can be found using the convex hull method. This will help you calculate the center of your palm. The convex dial method means that all lines between two points on the inside of the hull are completely inside.

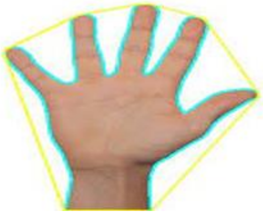


Fig. 4. Convex hull

The convex hull method uses shape contours to identify convex and concave (defective) points. These points give you an idea of the object's shape. In a hand, for example, there should be 5 protrusions (1 for each finger) and 4 defects (1 between two adjacent fingers).

By counting the number of convex and defect points, we can determine how many fingers the user is showing us. For example, if there are two convex and two defect points, the user is showing us two fits of anger. We used OpenCV functions like finding Contours in our implementation. The table below shows an example output for centroid detection and defect count. The next step is to draw a circle around your finger. The center point corresponds to the center of the palm and the radius is 70% of the maximum Euclidean distance between the center of the palm and the poles. This creates a convex hull, and when an object deviates from this hull, it becomes a convex defect.

An image can help us visualize what we're trying to achieve. In this example, we want to show the flow of a process from start to finish, so we'll use a line to connect the two points and then add a circle at the end to indicate completion.

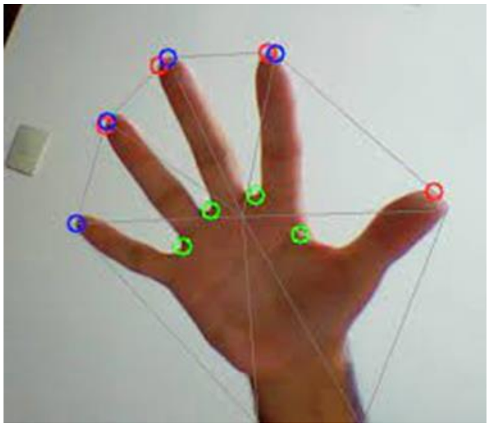


Fig. 5 Convexity defects

The law of cosines in trigonometry relates the length of each side of a triangle to the cosine of one of its angles. According to the law of cosines, where the angle formed by the sides of lengths a and b and the opposite side of length c. it should be less than 90 degrees. We consider gamma to be a finger if it is less than 90 degrees or pi/2.

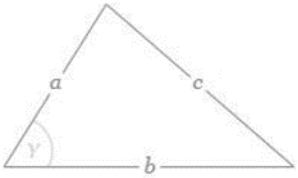


Fig. 6. Cosine rule triangle

3.6Formulas

1. $\gamma = \cos^{-1}\left(\frac{a^2 + b^2 - c^2}{2ab}\right)$

2. $c = \sqrt{a^2 + b^2 - 2abcos\gamma}$

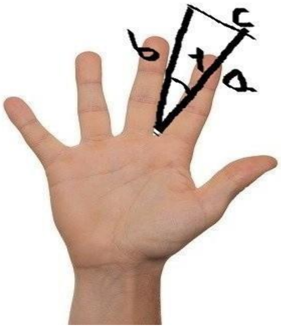


Fig. 7. Finding the angle between fingers

As we stated before, gamma (the angle between two intersecting lines) less than 90 degrees was determined to be a finger. After we found the gamma, we drew a circle with an approximate distance to the farthest point from the centre. Then, Put text on the image to indicate the number of fingers raised (cnt). The process of distinguishing hand gestures is dynamic- meaning that once a gesture is completed, we go back to the first step and start over again.

4 Outputs

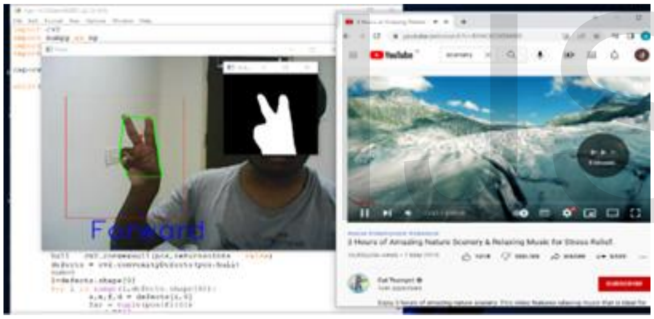
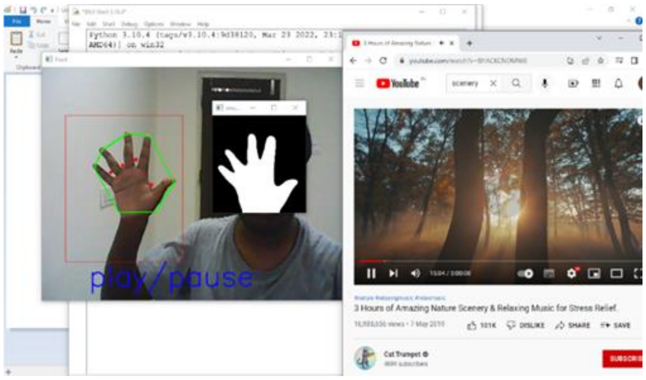

Forward
3 Hours of Amazing Nature Scenery & Relaxing Music for Stress Relief.

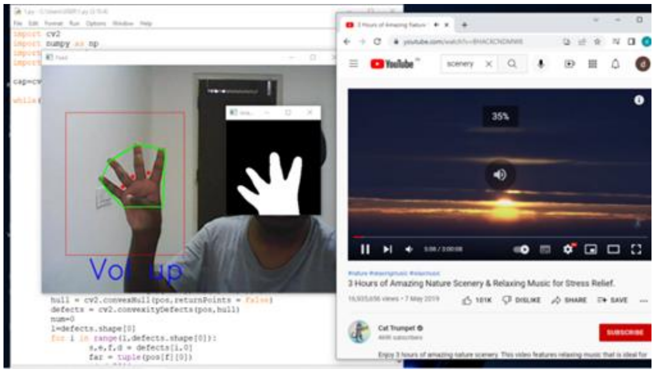
Fig. 8. Forward video



```
Python 3.10.4 (tags/v3.10.4:9d38120, Mar 23 2022, 23:1
AMD64)] on win32
```

play/pause
3 Hours of Amazing Nature Scenery & Relaxing Music for Stress Relief.

Fig. 9 play/pause video

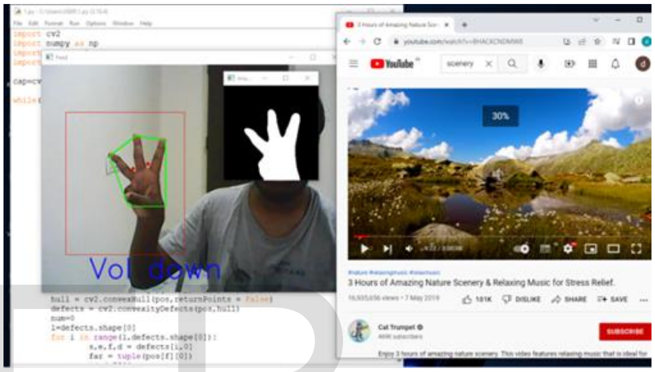


```
import cv2
import numpy as np
```

```
hull = cv2.convexHull(pos,returnPoints = False)
defects = cv2.convexityDefects(pos,hull)
num=0
l=defects.shape[0]
for i in range(1,defects.shape[0]):
        s,e,f,d = defects[i,0]
        far = tuple(pos[f][0])
```

Vol up
35%
3 Hours of Amazing Nature Scenery & Relaxing Music for Stress Relief.

Fig. 10 Volume up



```
import cv2
import numpy as np
```

```
hull = cv2.convexHull(pos,returnPoints = False)
defects = cv2.convexityDefects(pos,hull)
num=0
l=defects.shape[0]
for i in range(1,defects.shape[0]):
        s,e,f,d = defects[i,0]
        far = tuple(pos[f][0])
```

Vol down
30%
3 Hours of Amazing Nature Scenery & Relaxing Music for Stress Relief.

Fig. 11 Volume down

Image merging is the process of combining two or more images into one image by first enhancing the images. This single image is more informative and accurate than the source images.

5 Accuracy

TABLE 1
ACCURACY TABLE

Method	Accuracy
Particle filtering approach	90.6%
Applied filtering approach	87.5
Binary classification approach	90.6%
SVM approach	85.7
CNN approach	95.6

6 Conclusion

After careful analysis, the project team concluded that hand gesture recognition development using Python and OpenCV is feasible using hand segmentation and hand detection theory. In summary, the system can be said to achieve the following goals:

1. Build a complete system for detecting, recognizing, and interpreting hand gestures using computer vision and Python/OpenCV.

2. Create a library of numbers and sign language gestures that can be used with the system.

This project has been a resounding success and has met all of its goals. We are confident that this system will be a valuable tool for anyone looking to develop a similar system in the future. Future recommendations for this system include performing additional gestures that can be performed by users with different skin tones and palm sizes. The current system uses only the right hand where the region of interest is defined. Therefore, it is a desirable improvement of the technology to perform various gestures through computer manipulation using both hands. Additionally, a background subtraction algorithm can be used to improve performance.

7 References

[1] Liu K, Chen C, Jafari R, Kehtarnavaz N (2014) Fusion of inertial and depth sensor data for robust hand gesture recognition. Sens J IEEE 14(6):1898–1903.

[2] Gupta, H.P.; Chudgar, H.S.; Mukherjee, S.; Dutta, T.; Sharma, K. A continuous hand gestures recognition technique for human-machine interaction using accelerometer and gyroscope sensors. IEEE Sens. J. 2016, 16, 6425–6432. [CrossRef].

[3] Afza, F.; Khan, M.A.; Sharif, M.; Kadry, S.; Manogaran, G.; Saba, T.; Ashraf, I.; Damaševiˇcius, R A framework of human action recognition using length control features fusion and weighted entropy variances-based feature selection. Image Vision Comput. 2021, 106, 104090. [CrossRef].

[4] Mitra, S.; Acharya, T. Gesture recognition: A survey. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 2007, 37, 311–324. [CrossRef]

[5] C. Von Hardenberg and F. Bérard 2001 Bare-hand human-computer interaction (Proceedings of the 2001 workshop on Perceptive user interfaces) pp 1–8.

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

[7] 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.

[8] Ahuja, M.K.; Singh, A. Static vision-based Hand Gesture recognition using principal component analysis. In Proceedings of the 2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE), Amritsar, India, 1– 2

[9] Itseez 2017 Open Source Computer Vision Library i tseez2015opencv (Online)Available: https://github.com/itseez/opencv (Accessed: 13-Oct-2017)

[10] Dadgostar, F., Barczak, A.L., Sarrafzadeh, A.: A colour hand gesture database for evaluating and improving algorithms on hand gesture and posture recognition. Res. Lett. Inf. Math. Sci. 7, 127– 134 (2005).

[11] S. M. M. Roomi, R. J. Priya, and H. Jayalakshmi 2010 Hand Gesture Recognition for Human-Computer Interaction (J. Comput. Science vol. 6) no. 9 pp. 1002–1007

[12] B. Ansari and S. Saic. ": A bibliography on blind methods for identifying image forgery." In: SignalProcess. Image Commun 25 (2009).

[13] R. Lockton 2002 Hand gesture recognition using computer vision (4th Year Proj. Rep.) pp 1–69

[14] P.S.Neethu, R.Suguna, Divya Sathish," An efficient method for Human Hand Gesture Detection and Recognition using Deep Learning Convolutional Neural Networks", Springer-Verlag GmbH Germany, part of Springer Nature 2020, March 2020.

[15] Chaikhumpha T, Chomphuwiset P (2018) Real-time two-hand gesture recognition with condensation and hidden Markov models. In: 2018 international workshop on advanced image technology (IWAIT), Chiang Mai, 2018, pp 1–4

[16] Symeonidis, K.: Hand gesture recognition using neural networks. Neural Netw. 13(5), 1 (1996)

[17] Zhigang, F. Computer gesture input and its application in human-computer interaction. Mini Micro Syst. 19996, 418–421.