A Game of Rock-Paper-Scissors Via Hand Gesture Recognition

Paresh Patil (pap408), Praphulla Bhawsar (pmb418), Rajat Panchal (rvp278)

Abstract

Hand tracking and extraction of hand shape features allows for hand gesture recognition, which can be used as a form of input for Human Computer Interaction. This is not only more intuitive but also faster in terms of input time, and there have been several methods that have been proposed to allow for efficient real-time gesture recognition. Our approach towards this was to implement the algorithm proposed by Lee et al. [1] for hand tracking using angles and contour detection. Beginning with skin detection to separate the hand from the background, we use a Canny-based mechanism to detect contours and later apply polygon formation with apex contour vertices to create a convex hull and determine defects and cavities. The hull and defects, calculated with certain thresholds in place, are then used to essentially understand the form of the user's hand via the number of fingers held up, and thus determine the sign being made. The end result is an application that uses the user's webcam to track his movements and play a game of rock-paper-scissors with the user in real-time, using only image processing techniques on the webcam video and without any learning or external reinforcement necessary.

1. Introduction

In recent years, there has been a significant effort devoted to gesture recognition and related work in body motion analysis due to a renewed interest in a more natural and immersive Human Computer Interaction (HCI).

Recognizing hand gestures, however, involves capturing the motion of a highly articulated human hand with roughly 30 degrees of freedom (DoF). Complete hand motion capturing involves finding the global hand movement and local finger motion such that the hand posture can be recovered.

On the other hand, vision-based extraction of the hand features has studied in measuring hand shape as well as a field of sign language recognition. There have been techniques proposed to

obtain hand shapes by matching hand silhouette with hand 3D CG in a simple background [7]. Sutherland et al. [8] use an architecture in which a supervisor selects and activates visual processes. They use image differencing and normalized histogram matching for hand tracking. To track hands, some techniques [9] used 2D deformable models and genetic algorithms. 3D information [6] has also been commonly used to track hands. Template matching has been a common approach for tracking and action recognition.

Our goal, however, is much simpler, since we need concern ourselves with only very few motions of the human hand, and very specifically, those of the fingers. The major issue to lookout for is to make the gesture recognition doable in real time while being sufficiently succinct to be run on a common personal computer. Most of the aforementioned methods generally rely on a greater availability of either time or computational power, and therefore cannot be used directly in our case.

We therefore take up the much simpler, albeit less robust and reliable, method of applying contour detection and hull creation to detect the sign made by the user. Since our scope was fairly limited, that of recognizing if the sign the user made was one of three - rock, paper or scissors. The fact that these three hand signs are quite easily distinguishable makes this task slightly easier, although doing it in real-time still poses a problem. In the following sections, we outline how we solved this issue and developed a gesture recognition that was limited but accurate and fast.

2. Methods and Implementation Details

In order to ensure fast gesture determination, it was necessary that the whole image not be processed, and instead there be a certain measure of relevance that could be used to extract only the necessary regions from the image. This could be thought of as a preprocessing step for each frame, in which we use a skin segmentation algorithm outlined in [6] to use the HSV color space and extract regions of the image of the skin tone. Fig 2.1 shows a typical frame captured by the program for gesture recognition.

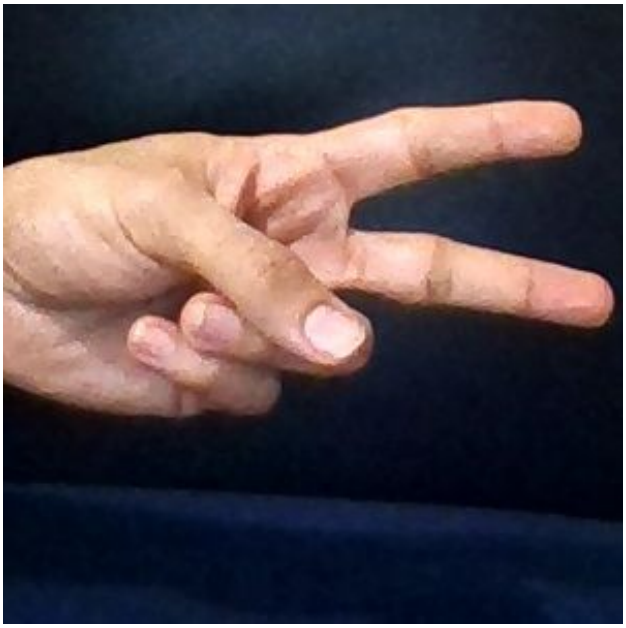


Fig.2.1 A typical gesture made by the user. The sign for 'Scissors'

2.1. Human Skin Segmentation

Human Skin detection deals with the recognition of skin-colored pixels and regions in a given image. Skin color is often used in human skin detection because it is invariant to orientation and size and is fast to process. We have implemented a skin detection algorithm wherein, The parameter for skin color detection is the HSV (Hue, Saturation, Value) color model. Skin detection is the process of finding skin-colored pixels and regions in an image or a video. This process is a pre-processing step to identify hand gestures.

The primary key for skin recognition from an image is the skin color. But color cannot be the only deciding factor due to the variation in skin tone according to different races. Other factors such as the light conditions also affect the results. Therefore, the skin tone is often combined with other cues like texture and edge features. This is achieved by breaking down the image into individual pixels and classifying them into skin colored and non-skin colored. One simple method is to check if each skin pixel falls into a defined color range or values in some coordinates of a color space. There are many skin color spaces like HSV, YCbCr, YIQ, YUV, etc. that are used for skin color segmentation. However we have used the HSV color space for skin segmentation.

The following factors should be considered for determining the threshold range:

1) Effect of illumination depending on the surroundings.
2) Individual characteristics such as age, sex and body parts.
3) Varying skin tone with respect to different races.

4) Other factors such as background colors, shadows and motion blur.

The algorithm implemented converts the entire image in a two dimensional matrix in which the column and row size is defined by the width and height of the image respectively. Once the frame is divided, each entry consists of a pixel of the frame.
The ARGB color of that particular pixel is determined. The ARGB value retrieved from the image for each pixel is a 32-bit value.
Bitwise AND operation was applied on these calculated values in order to extract only the bits corresponding to that particular color. This is applied to each and every pixel of the image. In order to make the recognition more precise the ARGB value is converted to HSV. The HSV value of each pixel is compared to the standard values of a skin pixel and decision is made whether the pixel is a skin pixel or not depending on whether the values lie in a range of predefined threshold values for each parameter. The ranges for a skin pixel used by our algorithm are as follows: 0.0 <= H <= 0.15 and 0.23 <= S <= 0.5.

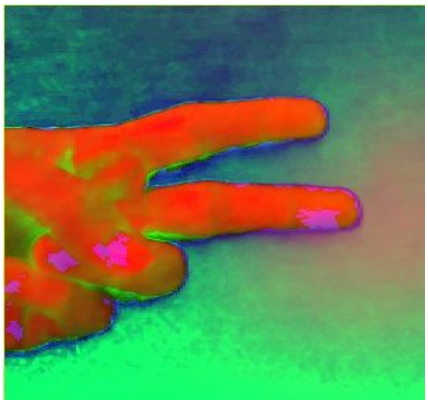


Fig2.2 HSV Segmented image

Fig2.3 Binary Image

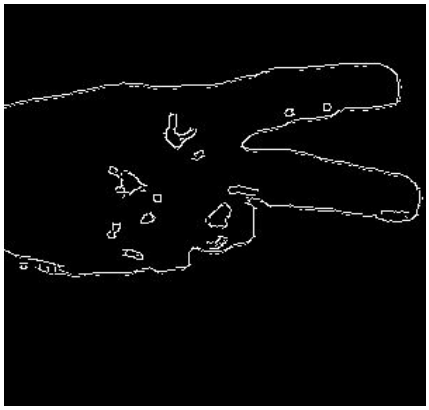


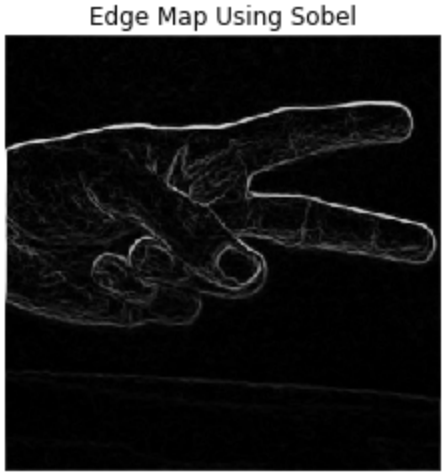
Fig2.4 Canny Edge Map

2.2. Contour Detection

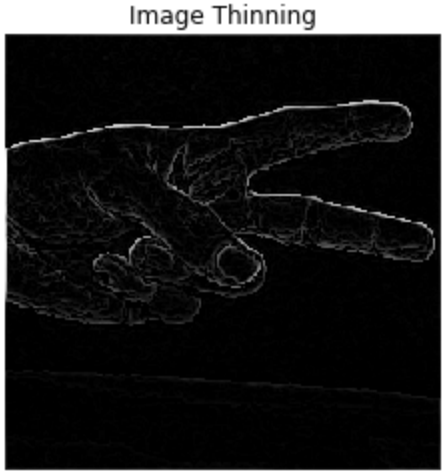
Contour detection has long been a fundamental problem in computer vision.Early approaches include the Canny edge detector, which was tuned to respond to sharp discontinuities in image brightness. Contours convey key information about object shape, and can be used to determine the complete geometry of the object itself. In the case of the hand, despite the fact that we needed to find just an approximate outlined form to create a convex hull around it, contour detection was not an easy task, primarily because contour detection is conventionally a processing heavy task.

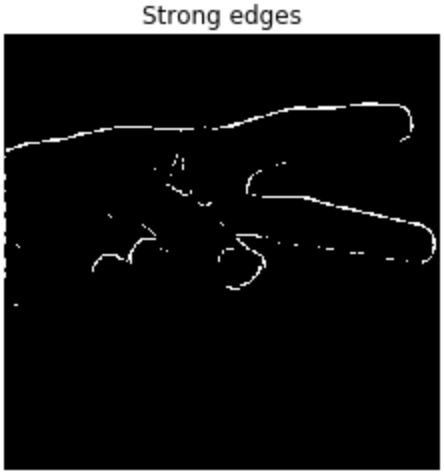
We did extensive research on multiple approaches, such as by using a generalized Hough Transform [11], calculating first, second and third order image moments, segmentation using superpixels and fuzzy k-means, and even using the more recent Morphological Snakes [4], but real-time contour detection was a real challenge. This was amplified by the fact that the project

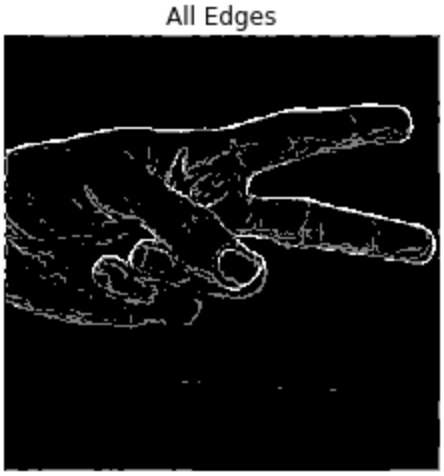
was implemented completely in the Python language, which is considerably slower than its lower-level counterparts such as C/C++. The results of our experiments can be seen in figures

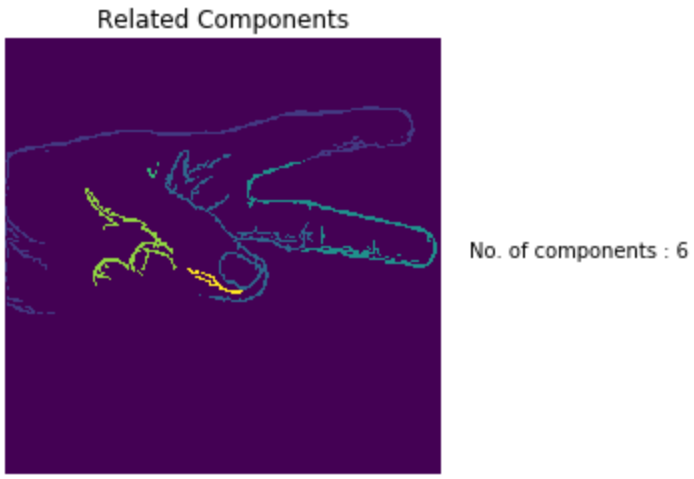


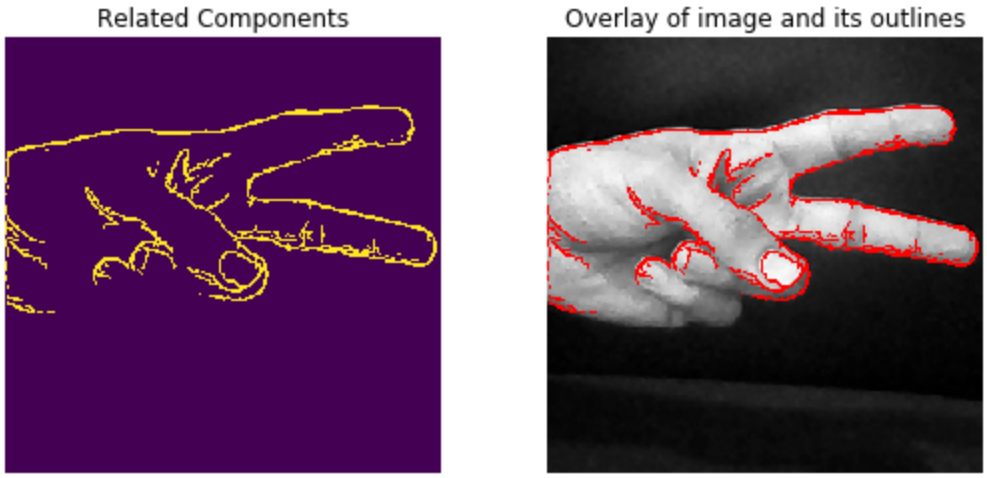
below:











2.3. Convex Hull Generation

After the contour points are obtained, the next step is to find a convex hull which completely encloses the subject's hand region. This was implemented using the monotone chain algorithm which is relatively efficient and straightforward.

Given a set of 2D points (contour points), a polygon enclosing the set of points is to be constructed such that no points are outside the polygon. The basic strategy of this algorithm is to determine the leftmost and rightmost points (extremes in terms of the x co-ordinate) and from those points determine the set of points that have the highest angle between them or that are the most clockwise in nature (done by determining the cross product) and in doing so we will obtain the points that constitute the convex hull.

The above simplification can be written as a pseudo code as follows –

Step 1: Sort the contour points with respect to their x coordinates, if there's a tie between two points then sort them with respect to their y coordinates (Lexicographic sorting).

Step 2: Initialize two empty lists which will constitute the upper and lower points of the convex hull points respectively.

Step 3: From the beginning of the sorted points determine, add the first two points to the list, the set of points which keep going clockwise (set of three points whose cross product is less than zero) and

add them to the list. This will determine the upper part of the convex hull.

Step 4: Repeat the above step but with reversed sorted points and the lower part of the convex hull will be determined.

Step 5: Join the two lists and the resulting list will contain the points that constitute the convex hull.

3. Results

The application made using the aforementioned methodologies is seen to be extremely real-time, and exhibits an accuracy of 77%, which would be impressive, but considering its fairly limited scope is only a minor achievement. It also requires a generally clutter-free background, since the contour detection technique used is quite simplistic, and having clutter in the background tends to create unexpected contours within the image.

Overall, the goal of creating an application to recognize a limited set of hand gestures in real-time was achieved. This can be seen in the video demonstration attached with the project submission.

4. References:

1. Lee, D., Lee, S.G.. Vision-based finger action recognition by angle detection and contour analysis. ETRI Journal 2011; 33 (3):415–422. doi:10.4218 / etrij.11.0110.0313.
2. Lee, Y.J., Lee, D.H.. Research on detecting face and hands for motion-based game using web camera. Proceedings - 2008 International Conference on Security Technology, SecTech 2008 2008;:7–12doi:10.1109 / SecTech.2008.14.
3. Lee, D., Park, Y.. Vision-based remote control system by motion detection and open finger counting. IEEE Transactions on Consumer Electronics 2009; 55 (4):2308–2313. doi:10.1109 / TCE.2009.5373803.
4. Márquez-Neila, P., Baumela, L., Álvarez, A morphological approach to curvature-based evolution of curves and surfaces, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 2013.
5. Zhou Hang: “An OPS Hand Tracking Algorithm Based on Optical Flow”, Software Engineering, 2009. WCSE '09. WRI World Congress on
6. Human Skin Detection Using RGB, HSV and YCbCr Color Models, S. Kolkur, D. Kalbande, P. Shimpi, C. Bapat, and J. Jatakia
7. R.Liu, B.Fang, Y.Y Tang, J. Wen and J. Qian, A fast convex hull algorithm with maximum inscribed circle affine transformation, Neurocomputing, Vol. 77 ,2012.
8. A.B. Smith, C.D. Jones, and E.F. Roberts, "An OPS Hand Tracking Algorithm Based on Optical Flow", Journal, Publisher, Location, Date, pp. 1-10.
9. J. Han, G. Awad, and A. Sutherland, "Automatic Skin Segmentation for Gesture Recognition combining Region and Support Vector Machine Active Learning", Proc. of IEEE FG'06, 2006, pp.237-242.
10. Kok F. Lai, Roland Chin, “Deformable Contours: Modeling and Extraction”