A study of Skin Colour Hand Segmentation and Finger tip detection

Vishal Vijayan (Author)
Dept. of Information Technology
SRH Hochschule Heidelberg
Heidelberg, Germany
Vishal.vijayan123@gmail.com

Prof Dr Ing Christof Jonietz (Author)
Dept. of information Technology
SRH Hochschule Heidelberg
Heidelberg, Germany
christof.jonietz@srh.de

Abstract—**The goal of this project is to grasp the deep knowledge about image processing and get hand on expertise in Opencv. The algorithm is used to perform skin color hand segmentation. Calculation is utilized to perform Fingertip detection. This undertaking can be used for securities and verification purposes.**

Keywords-component; skin colour hand segmentation; fingertip detection;

I. INTRODUCTION

Digital image processing is the use of computer algorithms to perform image processing on digital images. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of multidimensional systems.

Fingertip detection is very popular and effective way used for human machine communication. It has been used in many applications including embedded systems, vision-based systems and medical applications. In hand gesture recognition, fingertip detection is an important part if image base models are being used. It faces many problems in skin segmentation due to luminance and intensity in images. Here we are focusing on fingertip detection of natural hand with real time performance. User can demonstrate the hand to framework confronting the palm to the camera though the bearing of hand isn't confined. User is allowed how hand in any direction as naturally hands move.

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. A HSV color space based skin filter would be used on the current image frame for hand segmentation. The skin filter is used to create a binary image with black background. This binary image would be smoothened using Gaussian blurring. Gaussian blurring is used to image smoothing and noise reduction.

Identify applicable sponsor/s here. If no sponsors, delete this text box. (sponsors)

II. SKIN COLOR HAND SEGMENTATION ALGORITHM

Skin detection is the process of finding skin-colored pixels and regions in an image or a video. This process is typically used as a preprocessing step to find regions that potentially have human faces and limbs in images. Skin image recognition is used in a wide range of image processing applications like face recognition, skin disease detection, gesture tracking and human-computer interaction. 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. 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 RGB, HSV, YCbCr, YIQ, YUV, etc. that are used for skin color segmentation. The skin detection is influenced by the parameters like Brightness, Contrast, Transparency, Illumination, and Saturation. The detection is normally optimized by taking into consideration combinations of the mentioned parameters in their ideal ranges.

A. Color spaces

A color space is a specific organization of colors. In combination with physical device profiling, it allows for reproducible representations of color, in both analog and digital representations. A color space is a useful method for users to understand the color capabilities of a digital device or file. It represents what a camera can see, a monitor can display, or a printer can print, etc. There are a variety of color spaces, such as RGB, CMY, HSV, HIS.

B. Red, Green and Blue (RGB) color model

The main purposes of RGB color model is to representation and display of images in electronic devices like television, computer and it is used in photography. In this model red, green and blue lights are added together to reproduce variety of colors. It is a device dependent color model. Computers graphics cards and LCD's are using RGB color spaces. Any color can be made using these base colors. It is depending on

how much color is taken from each color. Also, if we do reverse operation we can break specific color into its red, green and blue. These values can be used to find the pixels in the image.

The choice of primary colors is related to the physiology of the human eye; good primaries are stimuli that maximize the difference between the responses of the cone cells of the human retina to light of different wavelengths, and that thereby make a large color triangle. [1]

The normal three kinds of light-sensitive photoreceptor cells in the human eye (cone cells) respond most to yellow (long wavelength or L), green (medium or M), and violet (short or S) light (peak wavelengths near 570 nm, 540 nm and 440 nm, respectively). The difference in the signals received from the three kinds allows the brain to differentiate a wide gamut of different colors, while being most sensitive (overall) to yellowish-green light and to differences between hues in the green-to-orange region [1].

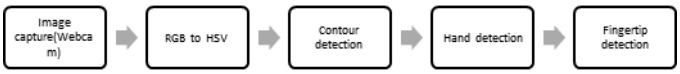
C. HSV model

HSL (hue, saturation, lightness) and HSV (hue, saturation, value) are two alternative representations of the RGB color model, designed in the 1970s by computer graphics researchers to more closely align with the way human vision perceives color-making attributes. In these models, colors of each hue are arranged in a radial slice, around a central axis of neutral colors which ranges from black at the bottom to white at the top. The HSV representation models the way paints of different colors mix together, with the saturation dimension resembling various shades of brightly colored paint, and the value dimension resembling the mixture of those paints with varying amounts of black or white paint. HSL and HSV are both cylindrical geometries, with shade, their precise measurement, beginning at the red essential at 0°, going through the green essential at 120° and the blue essential at 240°, and afterward wrapping back to red at 360°. In every geometry, the focal vertical pivot includes the nonpartisan, colorless, or dark hues, extending from dark at gentility 0 or esteem 0, the base, to white at lightness 1 or value 1, the top.
HSV shading space is better, in light of the fact that in there the data of shading is separated from the data of enlightenment. HSV remains for Hue (the color information data), S (Saturation, e.g., the level of 'color' present) and V (brightness, e.g., the level of 'white' shading present). For the most part, human skin lies amongst (H=0, S=58) and (H=50, S=173). With RGB the color will have values like (0.5, 0.5, 0.25), whereas for HSV it will be (30° , √3/4, 0.5). HSV is best used when a user is selecting a color interactively It is usually much easier for a user to get to a desired color as compared to using RGB [3].

D. Opencv

It is a library of programming functions mainly using in real time applications in computer vision. It was developed by Intel, later supported by Willow garage. The main application of OpenCV is 2d and 3D application, facial recognition system, Gesture recognition and Human computer interaction. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface [4].

III. IMPLIMENTATION



A. Image capture

Here we are going implementing our detector. The first thing we need is to read the image from the webcam. OpenCV contains a module (imgproc) responsible for capturing images and videos. Using this we can do image processing like Geometric image transformations, Histograms, shape descriptors and image filtering. In a loop we are taking images in a interval of 30 milli seconds. The argument passed to the "VideoCapture" object indicates the index of camera that we want to access. the argument of "waitKey" function, which waits x milliseconds until a key has been pressed and returns -1 if none was pressed. Using "imshow" function captured image can be displayed.

Image captured using webcam.

B. HSV conversion

Now we must convert captured frame which is in BGR color space to HSV and then segment the color using "inRange" operator. For color segmenting, the HSV color space is much better, because in there the information of color is dissociated from the information of illumination. This model discriminates luminance from chrominance. This is a more intuitive method for describing colors, and because the intensity is independent of the color information this is very useful model for computer vision. cvtColor(CV_BGR2HSV) is used for this operation. The converted image is then processed to detect the skin tone. Generally 0<H<0.25 - 0.15<S<0.9 - 0.2<V<0.95 .

The "inRange" function receives the image that we want to threshold, the interval lower bound, the interval upper bound and the output image. The result is a black and white image, where pixels that have values inside the interval are colored with white, otherwise black. This is poor segmentation. To get good segmentation we need to do fine tuning. For this I added track bars. Noise is also very high in this image. Toa void this we apply special techniques like median blur and dilate.

C. *Median blur*

Smoothing, also called blurring, is a simple and frequently used image processing operation. It is doing in order to reduce noise. To perform a smoothing operation, we will apply a filter to our image. The most common type of filters are linear, in which an output pixel's value (i.e. g(i,j)) is determined as a weighted sum of input pixel values (i.e. f(i+k,j+l))[5].

g(i,j) = \sum_{k,l} f(i+k, j+l) h(k,l).

The median filter run through each element of the signal (in this case the image) and replace each pixel with the median of its neighboring pixels (located in a square neighborhood around the evaluated pixel).

The function smoothest an image using the median filter with the aperture. Each channel of a multi-channel image is processed independently. In-place operation is supported.

D. Dilate

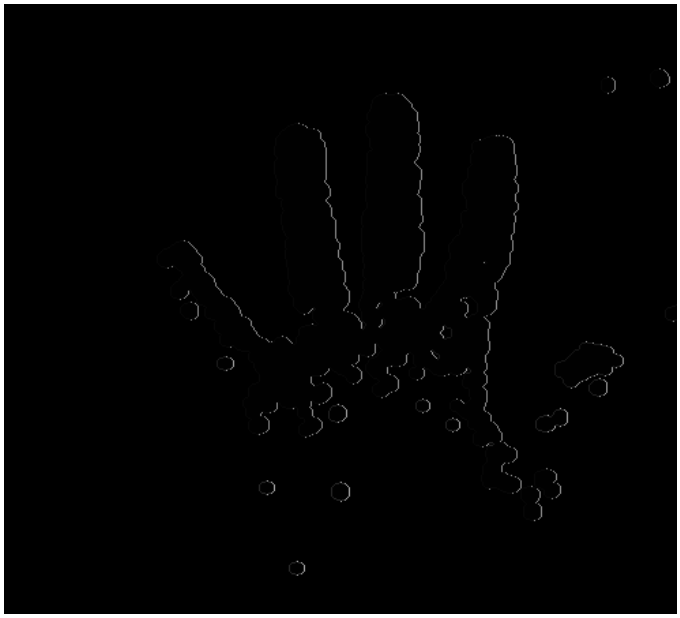
Two very common morphology operators Dilation and Erosion. They have wide uses including remove noise, isolation if individual elements and joining disparate elements and finding intensity bumps or holes in the image. Dilations operations consists of convoluting an image A with some kernel (B), which can have any shape or size, usually a square or circle. Erosion operation is the sister of dilation. What this does is to compute a local minimum over the area of the kernel. Despite working well this approach has many problems and limitation, if the background isn't static or if the illumination of the scene changes, it doesn't work.

E. Contour detection

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition. For better accuracy, use binary images. So before finding contours, apply threshold or canny edge detection. Here I am applying threshold before finding contour. In OpenCV, finding contours is like finding white object from black background. So remember, object to be found should be white and background should be black [6].

There are three arguments in cv2.findContours() function, first one is source image, second is contour retrieval mode, third is contour approximation method. And it outputs a modified image, the contours and hierarchy. Contours are often obtained from edges, but they are aimed at being object contours. Thus, they need to be closed curves. You can think of them as boundaries (some Image Processing algorithms & libraries call them like that). When they are obtained from edges, you need to connect the edges to obtain a closed

contour. cv2.drawContours function is used to draw the contours.

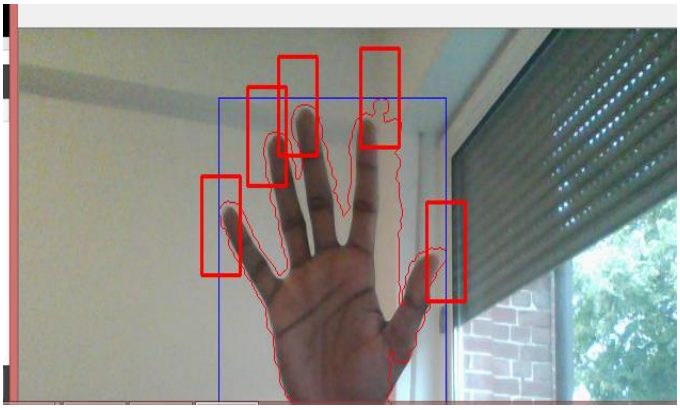


The “findContours” function expects to receive a vector of vector of points, or, in another words, a vector of polygons. There are many detection methods. Here I chose “CV_RETR_EXTERNAL”, which means it will just return the most external contour, ignoring eventual contours that are inside the most external region. I then compare the areas of the returned polygons (through the “contourArea” function) to get the largest and then draw it on screen (through the “drawContours” function).

F. Detecting fingerprints

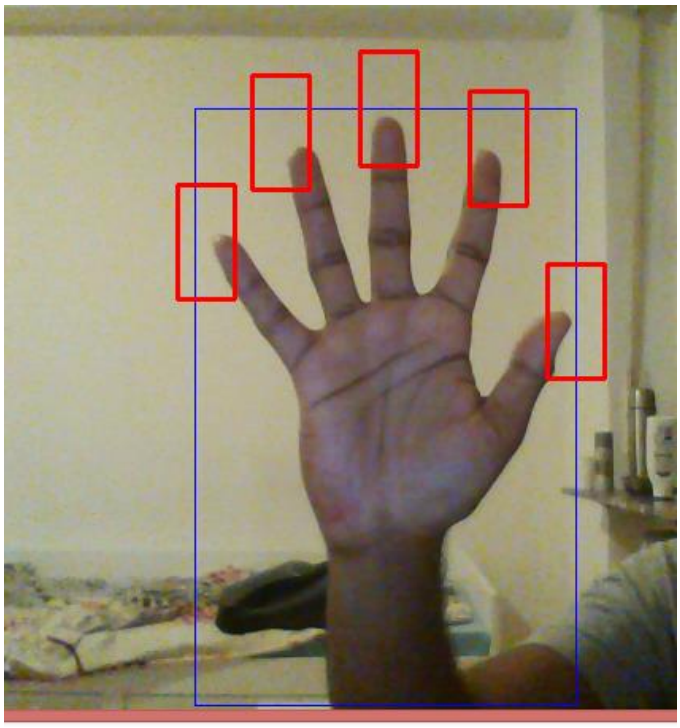
For getting the fingertip detection we are using convex hull technique. Convex Hull is the smallest convex set that contains a set of points. And a convex set is a set of points such that, if we trace a straight line from any pair of points in the set, that line must be also be inside the region. The result is then a nice, smooth region, much easier to be analyzed than our contour, that contains many imperfections. this algorithm is also implemented on OpenCV through the “convexHull” function.

It expects to receive a set of polygons like findcontours function. Here we are calculating the convex hull of largest contour. Next step is to find the tip of the finger. This can be detecting using convex hull region. We will get a convex hull polygon; their locations coincide with corners. Instead of doing manually we can use convexDefects function. There are “gaps” between the convex hull region and our contour region. The “convexDefects” will try to approximate those gaps using straight lines.



The “convexityDefects” function returns a vector of tuples of four values. The first value is the initial point of the defect region. The second value is the ending point of the defect region. The third value is the “middle” point of the defect region that connects the initial point and the ending point. The result is then two lines: One from the initial point to the

middle point and one from middle point to the ending point. In order to find the center of the contour, we must involve it with a bounding box. OpenCV already has a function for that called "boundingBox". We just are going to consider points that are between -30° and 160°.



Final output.

IV. CONCLUTION

In this paper a segmentation method is represented for detecting hands and detect fingertip from the image. The accuracy of this solution is not the always the best and the program may need to be tuned differently for different environments. The algorithm is capable of processing images of different light conditions such as brightness etc. The future scope of this algorithm is to detect face, hand as well as hand gestures which can be used for security purpose, aid for physically challenged (deaf) individuals or for skin disease detection. This software has been written with limited knowledge of both OpenCV and C++, there's probably plenty of room for optimizations and performance improvements

REFERENCES

[1] https://en.wikipedia.org/wiki/RGB_color_model

[2] https://en.wikipedia.org/wiki/HSL_and_HSV

[3] Poorani. M, Prathiba. T, Ravindran. G: "Integrated Feature Extraction for Image Retrieval", International Journal of Computer Science and Mobile Computing vol.2 no. 2, 28-35(February2013)

[4] https://en.wikipedia.org/wiki/OpenCV

[5] https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html

[6] https://docs.opencv.org/3.3.1/d4/d73/tutorial_py_contours_begin.html

[7] https://picoledelimao.github.io/blog/2015/11/15/fingertip-detection-on-opencv/