# Final project. Finger Detection

**Alex Carrillo and Robert Tura**
Image Processing and Artificial Vision

May 22, 2020
*Universitat Politècnica de Catalunya*

## 1 Introduction

The purpose of this report is to show an approach of using bidimensional histogram information of images in some color spaces to extract which regions of a picture contain human skin, and we also present which processing steps improve these predictions. Furthermore, we explain our idea on finger classification of those segmented images, *i.e.* how many fingers are shown, and we expose the results obtained, the difficulties and further work.

## 2 Color spaces

A color space is a method which describes the range of colors of an image as tuples of numbers, typically specified using 3 or 4 coordinates. These parameters represent the position of the color within a subspace as a single dot. In order to characterize a color for a particular application several criteria can be used, from a definition by its attributes of brightness, hue or colorfulness, to a specification in terms of the amount of red, green and blue light required.

### 2.1 RGB (and RG)

RGB is the most common color space for digital images and widely used for displaying images in electronic systems. This space is described as an *additive color model* in the sense that three light beams (red, green and blue) are mixed together to make a final color. Each one of these components can have an arbitrary intensity from fully off, so all components to zero give black color, to fully on, which for each component gives white. However, in computing, the component values are usually stored as integers between 0 to 255. Moreover, two important aspects of this color space must be stated:

(i) The RGB model has non-uniformity. This means that even though we display evenly spaced hue values side by side, the corresponding effect is not linear to the human eye.

(ii) Chrominance and luminance data is mixed. This means that pure color and light information can not be directly extracted by its components, but with some transformation.

For the reasons above, RGB is not suitable for color analysis and color-based recognition, in general. Nevertheless, this color space can be converted to a normalized space, called RG, that somehow makes the model less dependent on the variation of brightness, a characteristic which is required for a reliable skin model. The following formula describes this new color space:

$$R' = \frac{R}{R+G+B} \quad G' = \frac{G}{R+G+B} \tag{1}$$

Notice that RG is actually a mere normalization of RGB space and, the resulting image can be represented using only 2 bytes per pixel (as opposed to 3 bytes per pixel in RGB), as the $B'$ component can be calculated from the other two. This "loss of information" is what removes the lighting information of the image.

### 2.2 HSV - Hue Saturation Value

HSV is an alternative of the RGB model which aims to be more approximate to the way humans perceive and interpret colour by somehow represent the way paints of different colors mix together. Hue is the parameter defined as "color", saturation means the concentration and value is the amount of brightness.

We do not provide any closed formula as the computations from RGB to HSV are many and several methods exist. In fact, the cost of conversion to HSV is expensive compared to other color spaces. However, we considered it as a color space candidate as it is less sensitive to lighting changes than RGB and hue is invariant on different skin tonalities, which could be interesing for our purpose.

### 2.3 YCbCr - Luminance Chrominance

The YCbCr color space is commonly used in image processing, as the luminance information is represented by a single component, Y, and color information is stored as two components, Cb and Cr. Hence, it is a clear and explicit separation of luminance and chrominance. The following formula describes this color space:

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ C_b &= R - Y, \quad C_r = B - Y \end{aligned} \tag{2}$$

This color model seems to be very useful for skin detection as those separated channel allow to discard the uneven ilumination an image can have (Y component) and focus only in pure color information (Cb and Cr) which is much more relevant and stable for a skin model.

## 3 Reasoning of the approach

### 3.1 Skin detection

For the skin detection algorithm we follow a simple but effective approach, but with several nuances on the way. The idea is to use only the regions of the images which contain skin pixels (by means of their true masks) in the training set, after converting them to the designated color space. (Remark that we have built the pipeline to easily change between 3 color spaces when training and predicting.) Now, for every TR image we compute its bidimensional histogram using the two more relevant channels of each color space:

- Green and Blue channels for the RG space (because they were the ones that showed better results). However, using another combination did not change much the outcome.
- Hue and Value channels for the HSV space. Because as we said, the saturation is the only that varies between skin colours (i.e. it increases with darker skin), which we are not interested in.
- Red crominance and Blue crominance for the YCbCr space. For the reasons stated above of preserving skin color very well.

Next, we average all bidimensional histograms we have calculated for each TR image. The resulting histogram is what we take as our skin model.

Now, with this idea, the skin detection on validation images is straightforward. First, we must convert the targeted image to the same color space as the skin model. We came up with an approach that goes through all the pixels on the targeted image and, focusing only on the same two channels we used for generating the skin model, we generate a new image which contains the bidimensional histogram bin value for the current pixel value. I.e. pixel $(i, j)$ of the predicted skin image is set to $hist[a, b]$, where $a$ and $b$ are the actual values of the 1st and 2nd channels of the targeted image (converted to the color space).

This way, all the pixels which have no value inside the bidimensional histogram skin region are set up to 0 (being part of the background) and the ones with potentially skin parts are set to an arbitrary number of the hist bin value, which somehow tells us the probability or certainty of really being a skin pixel. With a simple normalization we map all pixels values between 0 and 1, and let a hyperparameter to choose the threshold with which pixels will be binarized. In our tests, a 0.95 was nice, meaning that we will only mark the pixel as skin if we are very sure of it.

The overall idea seemed to work well in many cases but we knew that with morphological tools, things could improve, and so it was as we will see in following sections. Taking into account that the algorithm predicted skin regions (marked as black) in a pointillism style we "merged" them by means of an *opening* with a 8x8 SE (generously sized) and we removed the remaining noise of wrongly detected skin in the background using a *closing* with a small 3x3 SE (no bigger than average noise). However, some bigger undetected regions inside the hands were not being merged and we used a hole filling technique (a closing by reconstruction with

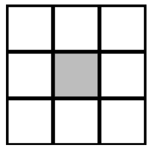

**Figure 1:** First Approach

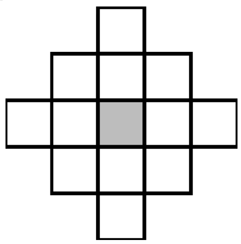

**Figure 2:** Second Approach

a white border background marker) which really made an improvement.

### 3.2 Finger classification

The first idea that came into our heads was a convolutional neural network because we have labeled data which is perfect for training a net, but the amount of data that we have is very low for a net to not be overfitted. The second approach we thought was the use of morphology to erase the hand of the bitmap and just keep the fingers in order to count them, but this was the basic idea that was proposed in the guidelines, so we went further.

Our method is based on the idea that fingers have a contour similar to a convex function in a certain window, so the method is based on following the contour and check, in a moving window, if this specific point is a "local maxima" with respect to the centroid of the contour. For this method to succeed we need to extract the contour of the masks and then map those points to a list of consequent points (See Figure 1 and 2), and then check one by one if they are the furthest point of the window.

## 4 Performance

### 4.1 Skin detection

After many tests and tweaks of the models, we got some positive results. Refering to the color spaces, RG and YCbCr performed really well both in training and validation, and the HSV space was by far the worst. We expected that because, when plotting the skin model histograms, HSV was the only which was generous with the skin pixel assignment range, while the other two comprised more restrictive regions. We think this is due to the presence of images with regions very similar to skin in the TR set, which HSV considered them very close. However, in summary the YCbCr model

| model | precision | recall | F-Score |
|---|---|---|---|
| | *Training* | | |
| RG | 87.63 | 96.10 | **91.67** |
| HSV | 62.86 | 96.42 | **76.10** |
| YCbCr | 91.01 | 93.04 | **92.01** |
| | *Validation* | | |
| RG | 88.70 | 81.47 | **84.93** |
| HSV | 52.82 | 90.50 | **66.71** |
| YCbCr | 96.50 | 91.39 | **93.87** |

**Table 1**

outperformed the F1-Score in the validation set, and we considered the best model, as well as it generated better human visible predictions as we will see.

### 4.2 Finger classification

We ended up with 4 different models. There are three aspects that change the model itself: adding a padding, the mapping of the contour, and the length of the sliding window. Our first model uses the first approach (Figure 1) for the line tracking of the mask contour and the second model makes a padding to ensure all contours are closed.

Then, when it comes to the third and fourth, we use the second approach for the line tracking (Figure 2); and the fourth model on top of that uses a smaller window: while the previous models use a window of 101 points the last one uses a window of 47 points. There are two metrics apart from the F-Score. First a simple accuracy based on the times our algorithm gets the amount of finger right, and also there is Finger Accuracy which tells us the accuracy finger-wise and not hand-wise, which is interesting because some times the issue we face is that the masks are not accurate enough and the contours have not the best of definitions and the line tracking algorithm fails.

| Model | Acc | Fin. Acc | F-Score |
|---|---|---|---|
| | *Training* | | |
| 8-Elem v1 | 5.66 | 11.98 | **11.21** |
| 8-Elem v2 | 5.66 | 14.19 | **12.32** |
| 12-Elem v1 | 15.81 | 38.27 | **28.68** |
| 12-Elem v2 | 17.39 | 54.01 | **31.28** |
| | *Validation* | | |
| 8-Elem v1 | 6.17 | 12.83 | **11.54** |
| 8-Elem v2 | 7.01 | 17.32 | **12.40** |
| 12-Elem v1 | 22.01 | 49.26 | **30.11** |
| 12-Elem v2 | 23.33 | 58.89 | **39.16** |

**Table 2**

As it is obvious, results are not that good as we would like, but some decent score are shown, regarding for instance the last model which perform better and has a fair F1-Score.

# 5 Results

Firstly, Figure 3 shows the pointillism effect generated by the prediction of skin. On the bottom-left we see the result after all the morphology processing versus the ground truth on the right.

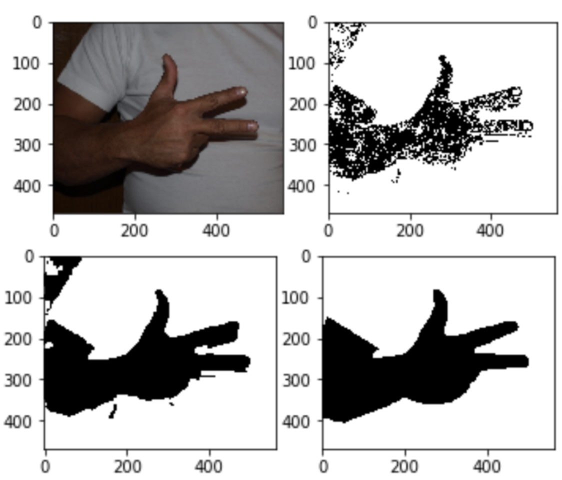

**Figure 3**

The results are quite amazing, and the predictions of our best model are not just good, but when correcting the images the improvement is even better, as we see in another example in Figure 4.

One last example is Figure 5 for which we will like to comment that, even though it seems an easy image to predict, we suffer a loss of nails (and it happened to many images) due to that those parts have not been detected enough and no morphological reconstruction is possible. This is not a huge problem for accuracy, but it definitely affects the quality of our finger classification algorithm.

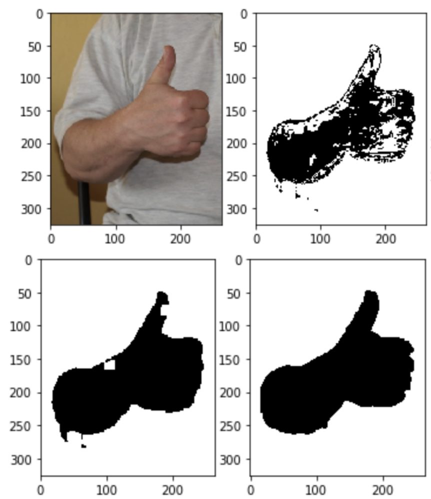

**Figure 4**

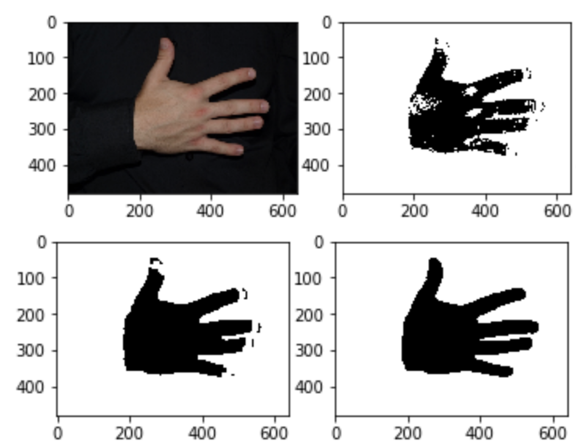

**Figure 5**

As we can see in the Figure 6, our algorithm ends up with the contour of the hand (mask) and also it gets from 1 to 6 points of a different color. This points are first of all the centroid of the object, and the rest are all the fingertips that are detected. (Zoom-in the images for a better view). The top-left image predicts 5 fingers when the hand was showing 4, that is because the end of the arm is the furthest point of its window, and that makes it a local minima recognizing it as a finger. The other two images predict the correct amount of fingers without any heavy problem, even though we did not get the contour to be closed.

One of the most common issues is the noise and all the wrong contours that are created with our own masks. That is, in essence, why our algorithm has an accuracy this low. With better masks (like the ones from the dataset) we would get better results.

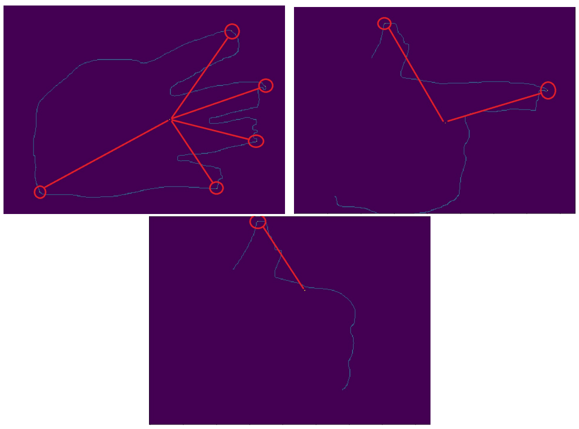

**Figure 6**

## 6 Parameter sensitivity

For the skin detection part, we had several parameter to tweak and it was not easy. However, in the process of optimizing the predictions we made an effort on the hyperparameter of the histogram binarizing threshold (presented in section 3.1). Firstly, as we did not know how color spaces would segment skin, we doubt if the regions of the histograms were decisive and, for so, we let that to be parameterized. For the hyperparameter, we made a round of tests and understood that it gave more or less certainty to select a pixel as skin. Therefore, we wanted to be near 1 (aligned with histogram bin counts), but, as we approached to it, we let to select pixels with a very low probability of being skin. In that range though, 0.95 worked better for us. Note we have not change the number of bins on the histograms in our tests, as we used a maximum of 255 and shortening them did not seem optimal at all.

Now, for the morphological filters part, we did not actually do a trial and error analysis, but rather applied common sense when determining the size of the structuring elements and the order of application of them, basically observing how the predicted images looked.

For the finger detection part we had two parameters that could be tweaked around. First of all, the size of the window that the algorithm checks for; there is a chance that we are looking at small neighborhoods and creating fake fingers, or that the neighborhood is too big, and therefore fingers are being neglected.

Also, we could change part of the algorithm that tracks the line of the contour, and also ensure the curves are closed. In our case, for some images there could be some rough edges that our simple approach does a poor job with.

## 7 Further improvements

In the case of detecting the skin, the results have exceeded our expectations, in our opinion, and we have not come up with other approaches that could be better in the period of time the work has been carried out. However, we emphasize the importance of better study a selected color space to be able to deeply understand what are the true qualities that make it a good skin model.

Our approach to the second part of the project was to be different and try to think out of the box with this algorithm. But the results did not meet the standards, so we could try to go for a morphological approach in order to make it improve.

## 8 Conclusions

Through this project, not only we have studied the theoretical aspects of color image detection and classifications algorithm but we have been able to implement it and perform several experiments as well. The fact of having had to implement it by our means and following our intuition, has lead us to a better and deeper understanding of how skin segmentation has to be addressed (and how different solutions can affect its precision, speed and accuracy) and what are the challenges of analytically classifying those images to detect fingers. In summary, taking into account the simplicity of the idea behind our approaches it has been very interesting to check its good validity.

## References

[1] Jorge Basilio et al. "Explicit image detection using YCbCr space color model as skin detection". In: Jan. 2011, pp. 123–128.

[2] Sandipan Dey. *Python Image Processing Cookbook*. Apr. 2020. ISBN: 978-1-78953-714-7.

[3] Noor Ibraheem and Rafiqul Zaman Khan. "Multiple Histogram Technique for Robust Skin Color Based Segmentation". In: *Journal of Engineering Research* 2 (May 2013), pp. 50–54.

[4] Kseniia Nikolskaia et al. "Skin Detection Technique Based on HSV Color Model and SLIC Segmentation Method". In: Nov. 2018.

[5] Emil Samuelsson. *Classification of skin pixels in images. Using feature recognition and threshold segmentation*. 2018. URL: https://www.diva-portal.org/smash/get/diva2:1278777/FULLTEXT01.pdf.

[6] Apostolos Tsagaris and Sotiris Manitsaris. "Color space comparison for skin detection in finger gesture recognition". In: *International Journal of Advances in Engineering  Technology (IJAET)* 6 (Sept. 2013), pp. 1431–1441.