Human Skin Segmentation for Hand Gesture classification

with thresholding based on YCbCr, HSV, RBG and CIEL*a*b* color spaces intersection and morphological operations in gray scale

Margarita Geleta & Oriol Nárvaez

May 10, 2020

Introduction

Skin segmentation is basic and key in the hand gesture recognition. Since skin color is within a threshold range, thresholding a color space, we can swiftly segment the desired region. However, there are two factors that have a direct influence upon this whole process:

- × Illumination conditions, noise effects and complex backgrounds.
- × Using a particular color space can influence image information representation [1].

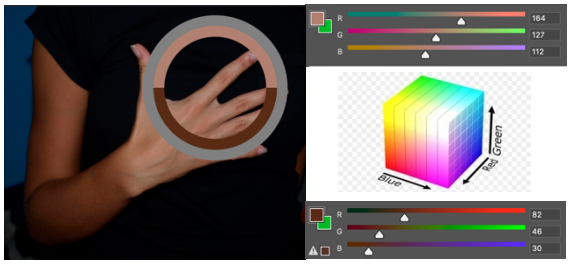
In this paper we propose two alternatives to approach the segmentation problem: (1) histogram-based and (2) color space heuristics.

After successfully segmenting human skin, we train a neural network for hand gesture detection to predict how many fingers are represented by the gesture and the performance is assessed with several metrics with respect to the segmentation hyperparameters.

Color spaces

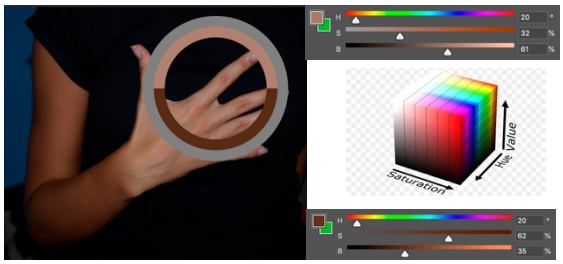
Skin segmentation is based on color rather than luminosity; thus, we need to select a color space which is robust to light variations. This is achieved if the color space separates well chrominance ("chroma", for short) from luminance ("luma", for short). We will review the color spaces we have used in this work, namely, YCbCr, HSV, RGB and CIEL*a*b*; and extract useful heuristics for skin detection.

Let's start form the basic RGB (Red, Green, Blue). The main drawback for skin segmentation of RGB is that is mixes luma with chroma, making not a favorable choice. Below we show the great change in range because of the light:



However, several heuristics can be extracting by observing human skin. Human skin color is closer to Red than to Green or Blue. Therefore, R is a dominant component and it has to be larger than the other two primitives. The heuristics are shown next: $R > 70$, $G > 40$, $B > 20$, $R > G$, $R > B$. Some works [2] propose using differences such as $|R - G| > 15$ for uniform daylight illumination and $|R - G| \leq 15$ for flashlight, but since this is luma dependent, we have neglected those rules, which actually did not lead to satisfactory results in our experiments.

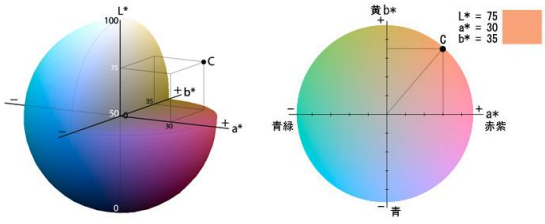
A more intuitive color space is HSV (Hue, Saturarion, Value). It is easier to understand because it separates luma from chroma, which is useful for segmentation.



As it can be seen, skin color is in the range of red-orange hue, so a direct constraint can be extracted: $H < 20$. To detect better dark skin, we observed that we had to add a lower bound $H > 0.0275$. Saturation is constrained to $S < 70$ and V has no restrictions, because if we threshold we will detect either light skin or dark skin.

Another color space which explicitly discriminates luma from chroma is YCbCr (Luma, blue-diffrence Chroma, red-difference Chroma). According to Chai and Ngan [3], the most representative ranges of chroma for skin are: $77 \leq Cb \leq 127$ and $133 \leq Cr \leq 173$.

Finally, the last color space which we have studied CIEL*a*b*. It also discriminates luma from chroma: the L primitive corresponds to human perception of lightness. We have not really found any deep-rooted heuristics in image processing in this color space, yet we have extracted our own.

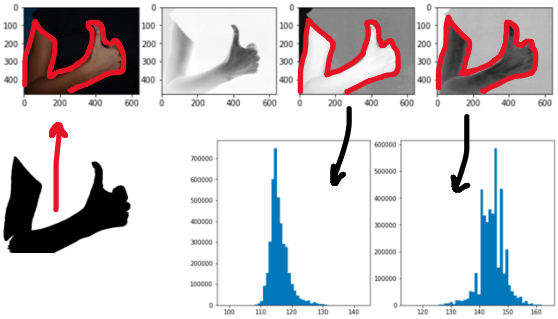


We have observed that skin color is located in the first quadrant of a*b* circle, which constraints us to $0 < a^* < 60$ and $b^* > 0.99$. The luma primitive is free, the reason is the same as for the V parameter in HSV. Also, we have noticed that the absolute difference between a* and b* does not exceed 12. Another useful formula has been derived, a quadratic one: $a^* - 0.009 \cdot b^{*2} \geq 0$.

Segmentation algorithms

As we have mentioned before, we are going to deal with two different approaches. The first one is the (1) histogram-based. The idea is the following: in the

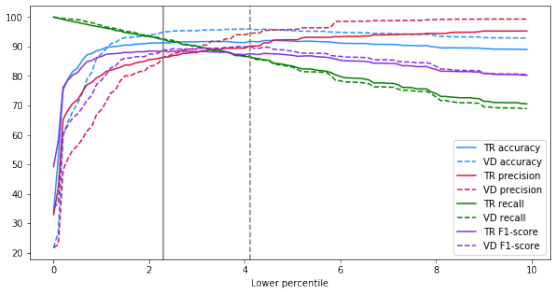
training dataset, we have pictures of hands and their corresponding masks. For each image, we extract the skin region. Then, we map the color values into a color space which discriminates well luma from chroma. We neglect the luma component and build two histograms for each of the chroma components. We take two percentile values for the tails of the obtained distributions which will act as lower and upper bounds. These are going to be our hyperparameters, along with the number of bins of the histogram. Finally, we threshold the original images in the chosen color space according to the lower and upper bounds of the chroma components.



Using our training dataset and the YCbCr color space, with 50 bins for chroma histograms, and using the 25%, 75% percentiles, the thresholds for chroma are: $122.12 \leq Cb \leq 121.21$ and $137.42 \leq Cr \leq 150.73$, which is in agreement with the thresholds explained in the previous section.

To assess the segmentation, we have introduced several metrics: accuracy, precision, recall and F1-score. Each of those metrics can be computed from True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) counts. TP can be computed by taking the intersection of true and predicted masks, TN by the intersection of the true and predicted background, FP by the intersection of the true background and the predicted mask and, finally, the FN by the intersection of the true mask and the predicted background.

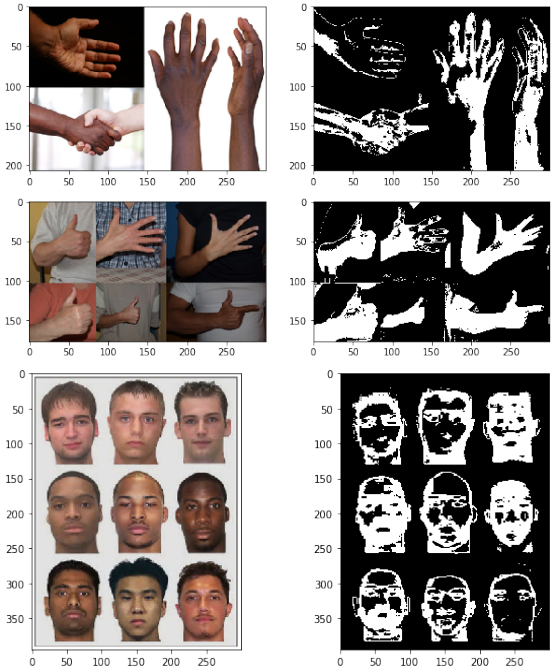
We have played with the hyperparameters, below we show a plot with the assessment metrics with respect to the percentiles' range without dilation.



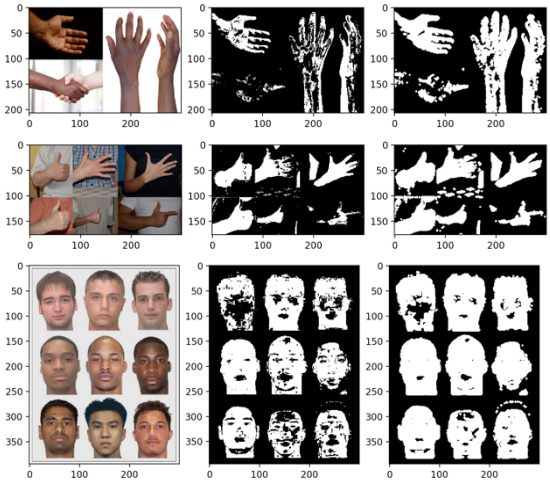
The vertical lines show where the maximum is attained in the F1-score metric (the straight line for training data and the dashed one for validation data). We can see that pattern for training and validation sets is the same and the best threshold range lies between [2.3, 97.7] and [4.1, 95.9].

Since this approach is data dependent, it induces bias. For example, it is uncapable of detecting faces effectively, which seems linked to different ambient light in the images. Also, we do miss some values from the recommended chroma range for YCbCr.

Next, we show some of the segmentations by the histogram-based algorithm.



Thus, our second approach (2) is based on all those color space heuristics mentioned in the previous section. Since we do not want to stick just to one color space, because we can miss precision, we decided to intersect all those color spaces. The algorithm is the following: for each image, we create a zero mask of the image dimensions. Then, we set a pixel to 1 if the value of the original image in that position is located in the intersection of the color spaces constraints. Otherwise, we leave it zero. This way, we obtain a mask for each image independently of the chroma distribution of the data.



Note that the first segmentation (center) is without dilation and the second segmentation (right) has a morphological dilation applied with a circular structuring element of size 2.

Some results

Just to compare both approaches, visually we can see that the second one makes it slightly better. Let's see some numbers. Histogram-based approach with 3%-97% percentile range and dilation with a structuring element of size 2 has given an accuracy of 94.88% ±

0.03%, a precision of 84.802% ± 0.12%, a recall of 94.612% ± 0.04% and an overall F1-score of 88.818% ± 0.08 on the validation set. Without dilation, the numbers change slightly, increasing the accuracy to 95.385% ± 0.03%, the precision to 89.281% ± 0.12 and reducing the recall to 90.2% ± 0.06%. Increasing the F1-score to 89.109% ± 0.08.

Now, the color space heuristics approach with a dilation with a structuring element of size 2 has given: an accuracy of 92.24% ± 0.075%, a precision of 91.216% ± 0.07%, a recall of 74.011% ± 0.28% and a F1-score of 78.171% ± 0.22%.

Our experiments can be summarized as follows: the histogram-based approach has shown to have high recall segmentation, with a high-variance precision; whereas the color heuristics method has shown dual results: high precision segmentation, with high-variance recall. Let's find out which of those methods will give us the best results in hand gesture recognition.

Convolutional network

In this phase, the objective was to find a classification model to correctly identify how many fingers are shown in each image.

For our case we have chosen a CNN – Convolutional Neural Network, because we thought it could give good results in image classification.

The model only has one convolutional layer since it would be better to keep it simple and the results would be good enough, but in the future, we could expand the model to multiple convolutional layers. We have also added a max pooling layer followed by two linear layers or dense layers.

The images are pre-processed with the code of the previous phase (can be found in `SkinDetector.py`), that binarizes them setting the pixels to 256 in areas in which it detects skin and sets to 0 the rest.

One important comment about the data is that we have a very small dataset of only 60 images to train and 45 to validate. With those few training samples we arrived quickly to >90% training accuracy, even hitting 100% in some occasions, but is important to say that training accuracy should not be the measure to evaluate different models, because it is possible that with 100% training accuracy the model is highly overfitted and will not perform well in the validation set, thus we have tweak some parameters to prevent overfitting.

In the final model, the pre-processing was made with the binarization based in color space heuristics and not based on histograms, we will comment later why this decision was made. Also, the final model has been trained with 20 epochs because we have observed that the recall was increasing with each epoch.

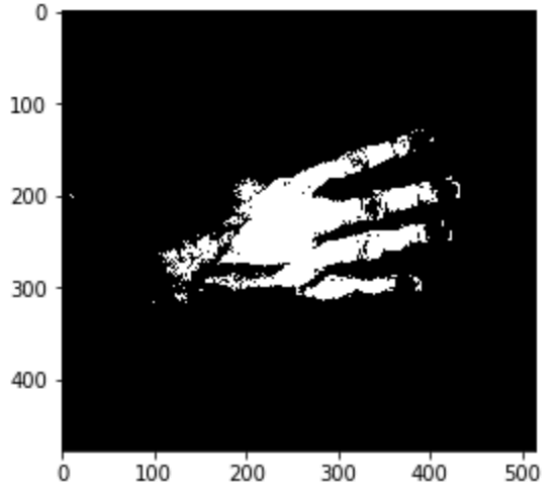
We have also added a dilation with a structuring element with size 2 to help with the skin detection. The CNN code is written in a python notebook and commented extensively to explain the different parts. It can be found in the `Classification.ipynb` file.

Results

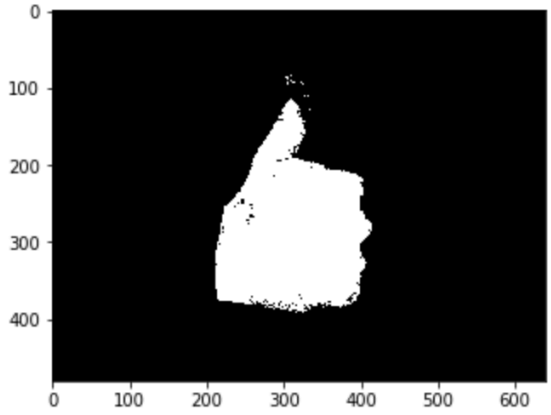
The model took about an hour to fully train the 20 epochs.

The chosen model performed considerably better than the other ones. In the test dataset the model obtains 76.44% accuracy and 87.77% recall with a final F1-score of 81.72%. We think that these results are quite great taking into account the few samples we had to train. Another model equally defined but without that dilation of size 2 obtains lower metrics, such as 73.33% test accuracy and 88,74% test recall. With a final F1-score of 80.30%.

More models were trained but with similar results as this last one, all of them were worse than the chosen one. Next, we show some test images and how the model has performed the prediction task:

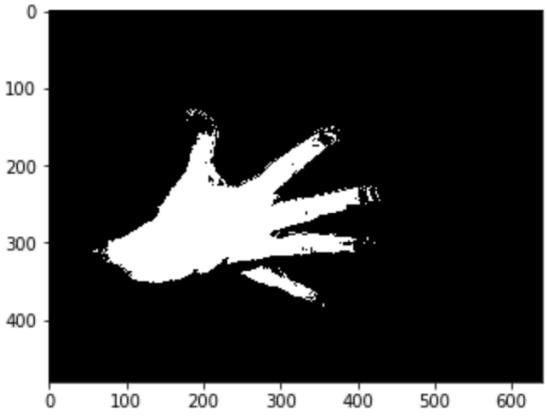


Here we can see one of the images and this one was predicted as 4, so this was correct.

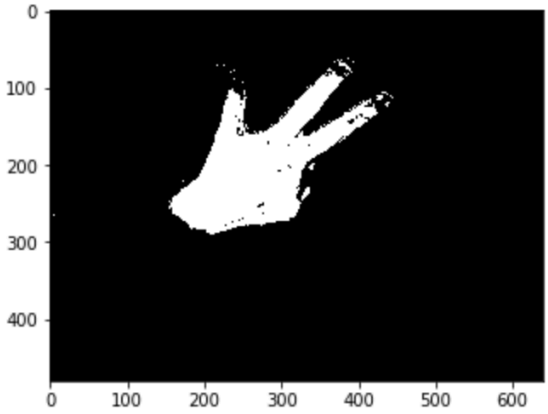


This one was predicted as 1, and as we can see this is also correct.

Now let's see some misclassifications.



This was predicted as 4, when we can see that it is really a 5.



And this was predicted as 2, when we can clearly see it should be a 3.

Actually, there is a point we would like to explain about the loss function or how, in fact, the model decides whether a prediction is well done or not. Obviously, it is not the same to predict that there are 4 fingers when the truth is 5 than when in fact there is only 1. The first case should have a higher score than the last one. It is not the same if you predict something closer to the reality than something completely wrong.

Further research

We have also tried to use the reconstruction operator. Our initial hypothesis was: if we can segment partially well the region corresponding to skin, then if we reconstruct from that region the whole skin region, we could obtain a better result. The objective was to reconstruct the whole skin region. However, there are two issues: first of all, the images are colorful – reconstructing can be done in grayscale, but in different color spaces, it does not have a meaningful interpretation. Secondly, if we convert the images to grayscale, we lose the information of the chroma corresponding to the skin color, and reconstructing a grayscale image, we cannot isolate the background, which also gets reconstructed. The result after applying reconstruction was not satisfactory, and since the operator is idempotent, there were no further changes in applying it again. Those experiments can be found in `main.py` (section `reconstruction_experiments`).

Conclusions

To conclude, in this research paper, we propose a hand gesture detection and classification system based on the thresholding intersecting several color spaces or histogram-based system, later combined with a convolutional neural network to classify the gestures.

Regarding segmentation, we have extracted the region corresponding to skin color pixels, from light to dark, this way avoiding ethnicity bias, and allowing to segment skin regions from complex backgrounds and different illumination. Finally, we use morphology operators to process the segmentation results and improve the system performance according to assessment metrics.

Talking about the classification model, we think it performs quite well. We could have also added some layers to increase the performance but we decided to keep it simple, only one convolutional layer and we see that it obtains good results such as +80% F1-score. Given the small amounts of training data that we had we think that our simple model has performed well.
We could have also used an SVM – Support Vector Machine classifier, but we thought that a CNN would be better since we can really fine tune the parameters of each layer to make it perform the way we want and extend it to multiple layers.

References

[1] Ennehar, B. C., Brahim, O., Hiccham, T., AN APPROPRIATE COLOR SPACE TO IMPROVE HUMAN SKIN DETECTION, (2010).
[2] Kovac, J., Peer, P., and Solin, F., 2D VERSUS 3D COLOR SPACE FACE DETECTION, (2003).
[3] Chai, D. and Ngan K. N., FACE SEGMENTATION USING SKIN COLOR MAP IN VIDEOPHONE APPLICATIONS, (1999).