Skin Detection Technique Based on HSV Color Model and SLIC Segmentation Method

Conference Paper · November 2018

CITATIONS
5

READS
460

4 authors, including:

Kseniia Nikolskaia
South Ural State University
32 PUBLICATIONS **39** CITATIONS
SEE PROFILE

Nadezhda Ezhova
South Ural State University
7 PUBLICATIONS **10** CITATIONS
SEE PROFILE

Skin Detection Technique Based on HSV Color Model and SLIC Segmentation Method*

Kseniia Nikolskaia[1], Nadezhda Ezhova[1], Anton Sinkov[1], and Maksim Medvedev[1]

South Ural State University, Chelyabinsk, Russia
nikolskaya174@gmail.com,eris909@mail.ru,
sinkov_96@mail.ru,own77d@gmail.com

Abstract. The paper is devoted to new skin detection technique based on the HSV color model and SLIC segmentation method. The algorithm of skin detection is described. Experiments results are presented. The influence of training images on the skin detection is shown. New skin detection algorithm implemented in Python language using OpenCV library is described.

Keywords: Skin detection · HSV · SLIC Superpixels · Computer vision · Pattern recognition

1 Introduction

Skin detection is the process of finding skin-colored pixels and regions in an image or a video. Skin detection applications are used for personality recognition, body-parts tracking, gesture analysis and adult content filtering and etc [1].

When the standard *RGB* color space is used, the skin detection can be very difficult under conditions of variable lighting and contrast. Therefore, the input image must be converted to another color space [2–4] that is invariant or at least insensitive to lighting changes, such as *HSV* [5].

The implemented skin detector converts the image into required color space and then uses the image histogram to mark each pixel: whether it belongs to skin. Image pixels are grouped in superpixels using the *SLIC* clustering method [6]. Thus, advantages of our skin detector are high processing speed and invariance under rotation and lighting changes.

The main steps of skin detection in the image are:

1. download the input image;
2. convert image to *HSV* color space;
3. generate the image histogram;
4. apply classifier to determine the probability of a given pixel being skin-colored;

* The work was partially supported by the Government of the Russian Federation according to Act 211 (contract №02.A03.21.0011.) and the Council on grants of the President of the Russian Federation (contract SP - 5430.2018.5).

5. divide the image into superpixels;
6. paint out superpixels where sum of probabilities less than the limit.

As a classifier, the *Naive Bayes* algorithm was chosen. The choice is due to the facts that this algorithm is the most accurate [7–9] and a small amount of data is required for it's learning [10, 11].

SLIC algorithm was chosen for image segmentation. It had the fewest errors among those considered in [12], and also it is the fastest one [12, 13].

The novelty of this research is the unique combination of technologies for solving the problem of skin detection in the image.

The paper has the following structure. In Section 2, color model convertion technique is described. In Section 3, algorithm of image histograms generation is given. In Section 4, application of Naive Bayes classifier is shown. In Section 5 superpixel segmentation is conducted. In Section 6, effects of the training example on skin detection are considered and results analysis is given. In Section 7, skin detection algorithm implemented in Python language using OpenCV library is described. In Section 8, the results are summarized and directions for further research are outlined.

2 HSV Color Model

The first step of skin detection is to convert the input image to *HSV* color space as the most suitable for our research [14–16]. The *HSV* color model is a cylindrical representation of the standard *RGB* model [17]. *HSV* stands for *Hue*, *Saturation*, and *Value*. The *Hue* is measured in degrees and varies from 0 to 360. It forms the base color. *Saturation* and *Value* (brightness) determine the proximity to white and black respectively. In the basic model they vary from 0 to 100, but in the *OpenCV* library used in the detector, they vary from 0 to 255.

In order to convert the image from the *RGB* to *HSV*, each pixel in the image is subjected to the following transformation [5]:

- Preliminarily, the maximum and minimum of R, G, B values C_max and C_min should be found and their difference M is calculated;
- The *Hue* is calculated:

$$H = \begin{cases} 0, & C_{max} = 0, \\ 60 \times \frac{G-B}{M}, & C_{max} = R, \\ 60 \times \frac{B-R}{M} + 120, & C_{max} = G, \\ 60 \times \frac{R-G}{M} + 240, & C_{max} = B. \end{cases} \tag{1}$$

- The *Saturation* is calculated:

$$S = \begin{cases} \frac{M}{C_{max}}, & C_{max} \neq 0, \\ 0, & C_{max} = 0. \end{cases} \tag{2}$$

- The *Value* is calculated:

$$V = C_{min}. \tag{3}$$

Calculating the above values for each pixel, we get an image in the *HSV* color space.

3 Histogram

The second step is to generate the input image histogram [18]. The histogram [19] is a graph of the distribution of digital image elements with different saturation. The horizontal axis represents pixel property values, and the vertical axis represents the number of pixels. The image representation in the form of a histogram is just another way of displaying the image. It is worth of noting that the color histogram itself is not exhaustive in representing image's features but, nevertheless, it is widely used. In image processing it is used either in combination with others parameters, such as color moments [20], or in its modified version [21], or in its original form with a slightly different approach to data compilation [18].

Let's take an image of people of different nationalities and build a histogram (see Fig. 1). Later, the histogram for the Value will not be taken into account in order to remove the effects of lighting changes.

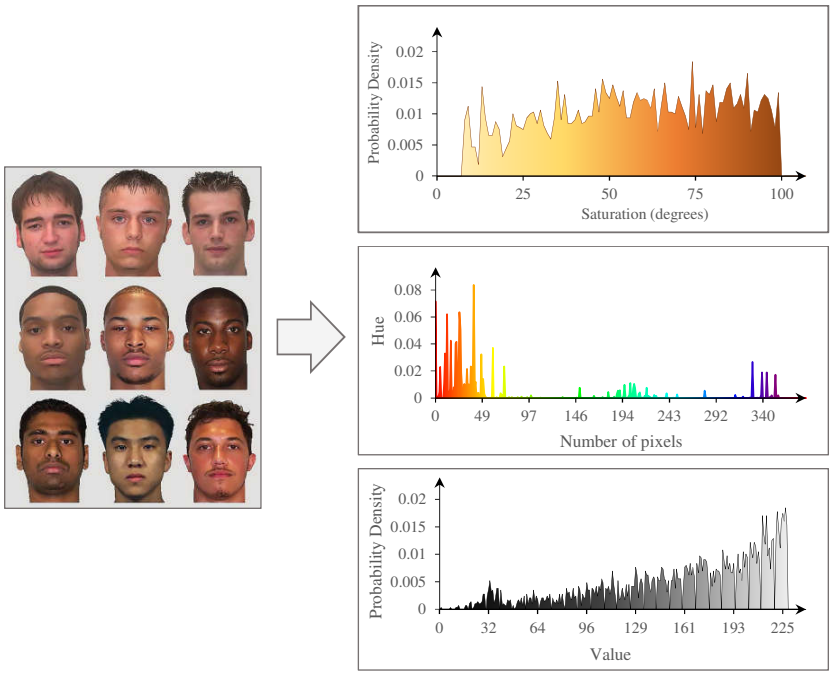

Fig. 1. Sample histogram

4 Classifier

The next step is to select a classification function [22] that will determine the probability of a given pixel being skin-colored. Such a function can be Bayesian network [23, 24], *Multilayer Perceptron* [25], *SVM* [26–28], *AdaBoost* [29, 30], *Naive Bayes*, *RBF network* [31] and etc.

In our detector the *Naive Bayes* algorithm is used. This method is based on the Bayes theorem with the assumption that the hypotheses are independent. In other words, the presence of any attribute in the class does not imply the existence of another. For example, if you consider an apple of green color, round in shape, and with a diameter of 5 sm, then all these parameters can be considered as independent, because it is possible to change one or more of them, but the object will still be an apple.

Naive Bayes algorithm is easy to implement. It is faster than many other algorithms [32]. Besides, despite very simplified conditions, *Naive Bayes* classifiers often work much better in many difficult situations [6, 33]. Also, the advantage of this algorithm is that it does not require a lot of training examples [34].

By the Bayes theorem, taking into account the variable y and the dependent characteristic vector $(x_1, \ldots, x_n)$, we obtain the following relation:

$$p(y|x_1, ..., x_n) = \frac{p(y)p(x_1, ..., x_n|y)}{p(x_1, ..., x_n)} \tag{4}$$

Using the assumption that the parameters are independent

$$p\left(x_i|x_{i+1}, \ldots, y\right) = p\left(x_i|y\right)$$

, the expression can be simplified:

$$p(y|x_1, ..., x_n) = \frac{p(y)\prod_{i-1}^{n} p(x_i|y)}{p(x_1, ..., x_n)} \tag{5}$$

for any i.

Notice that the variable $p\left(x_1, \ldots, x_n\right)$ can be eliminated from the original equation (4), because it is a constant:

$$p(y|x_1, ..., x_n) \propto p(y)\prod_{i=1}^{n} p(x_i|y) \tag{6}$$

Using the transformations described above, the classification rule can be obtained [35]:

$$Y = argmax_y p(y)\prod_{i=1}^{n} p(x_i|y) \tag{7}$$

This rule will be applied to each pixel. Thus, we obtain the *probability map (PMap)* [36] of the image (see Fig. 2).

Fig. 2. Probability map of the image

5 Superpixel Segmentation

Superpixel segmentation algorithms combine the pixels into the atomic regions, which will later be checked on being skin-colored. There are various algorithms for segmentation [13, 37]. One of them is *SLIC Superpixels* algorithm. It is an adaptation of the *k-means* method for superpixels [6]. The difference is that the algorithm is more efficient because the detection area is reduced to the size of a superpixel.

The algorithm is simple for understanding and implementation [12]. Initially, only the number of regions k is specified. The image is divided into k segments of size S, and center is located at the point with the lowest gradient. Then, for each pixel, a $2Sx2S$-sized area is considered and it is attached to the segment with the smallest difference in the distance D_s, calculated from the equation (8) [38]. After that, in each cluster, the center moves to the point with the lowest gradient and again all the pixels are rearranged [12, 4]. This occurs until a certain threshold of the minimum distance is reached.

$$D_s = d_{lab} + \frac{m}{s} d_{xy}, \tag{8}$$

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}, \tag{9}$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}, \tag{10}$$

where d_{lab} is the distance in *CIELAB* color space between $k-$ and i-pixels (color distance); d_{xy} is the distance between coordinates $k-$ and i-pixels (spatial distance); m is the parameter that affects the size of a superpixel. The more m, the more the spatial proximity affects the overall distance.

After all transformations of the image used in the previous section we get Figure 3 (for $m = 10$).

The dependence of the segmentation on the coefficient m is shown on Figure 4. It is clear, that the greater the m, the smaller the segments. The optimal is $m = 10$.

During the segmentation, unnecessary regions was removed and possible image noises were excluded. Thus, skin will be selected entirely, and the other pixels, which falsely have a high probability, will not be colored out. For comparison, in Figure 5 there are two methods results. It is noticeable that the left image is less informative. Segments were selected by threshold function, i.e. if the sum of the probabilities of the whole segment was greater than a certain

Fig. 3. SLIC superpixels segmentation

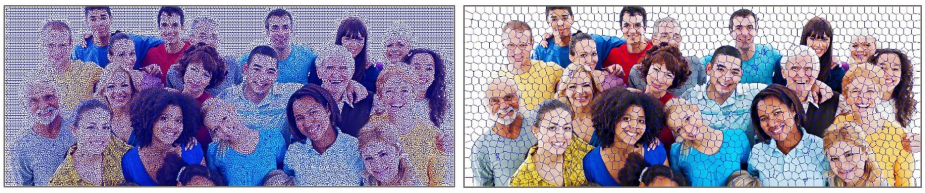

Fig. 4. Dependence of image partitioning on the coefficient m (right image $m = 5$, left image $m = 20$)

value, then the pixel was not colored out and recognized as the skin. The right image shows that not only the skin was colored out. This is due to the people of the training example are with hair, so the hair in Figure 5 was also recognized, as well as parts of the clothing, because its color is similar to the dark skin color (see Fig. 5).

Fig. 5. Methods comparison

6 The Effect of the Training Example on the Result

In this section, let us consider the dependence of the detection result on a variety of skin colors on the training example. Let us take the original image with people of different races (see Fig. 6) and test different training examples on it.

Fig. 6. Input image

Comparative figures are presented below (see Fig. 7, Fig. 8, Fig. 9, Fig. 10).

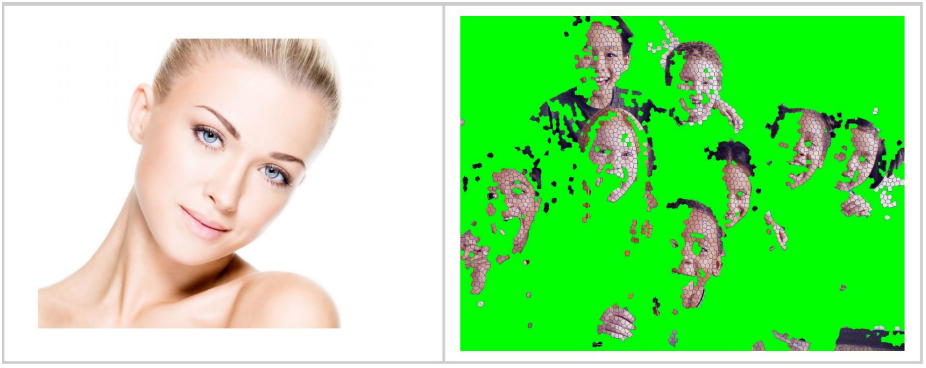

Fig. 7. 1st experiment

Experiments show that the image on which the histogram is generated significantly affects the skin detection. For the training images of people of different races should be chosen. The first and second tests are the examples of incorrectly selected training images. The most accurate result is obtained in the fourth test.

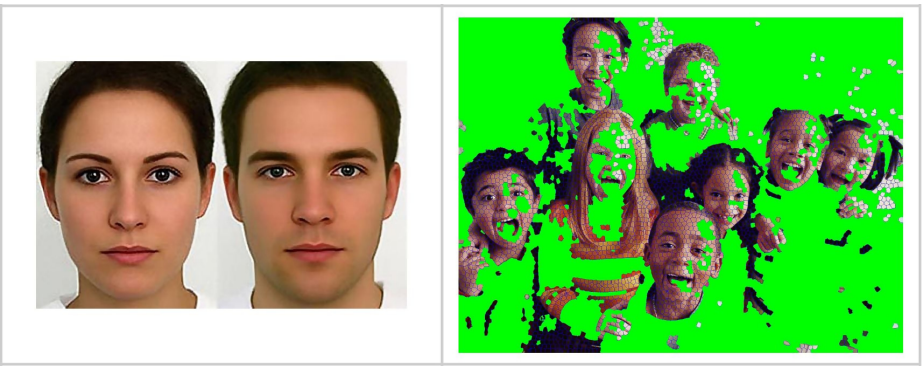

Fig. 8. 2nd experiment

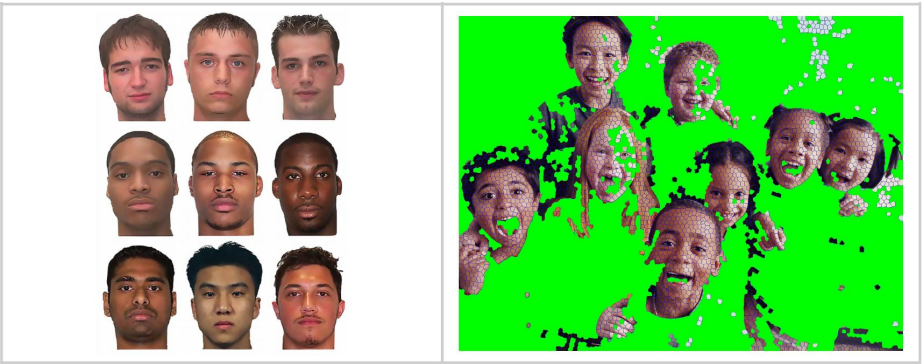

Fig. 9. 3rd experiment

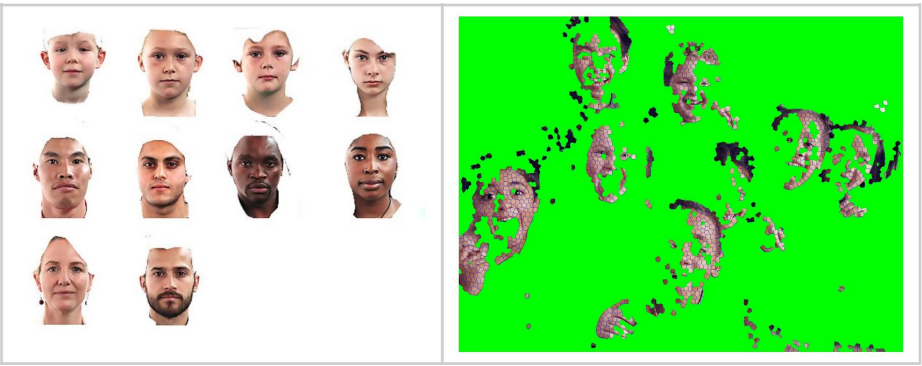

Fig. 10. 4th experiment

However, there are also false positives, for example, then the clothes color is similar to the skin color.

7 Skin Detector Implementation

The program is implemented in Python (v3.6.1) programming language. This language is chosen because it is commonly used for solving the pattern recognition problem, a large number of libraries are freely available - it greatly simplifies the development. The following libraries are used in the program:

- OpenCV (v3.2.0) is applied for the basic calculations;
- Numpy (v1.12.1) accelerates and simplifies operations with arrays [39], which OpenCV operates with [40];
- Matplotlib (v2.0.2) is used for plotting Numpy arrays as images [41].

To download input image the function *cv2.imread* used. It is worth noting that the downloaded image is in *BGR* color space [42] by default, so it should be converted from the *BGR* color space to *HSV* (see Fig. 11). To do this, we used the function *cv2.cvtColor*. Next, a mask of the training image is generated using the function *cv2.inRange*, which sets the lower and upper bounds of colors involved in generating the histogram. This step is needed to remove the background from the image. Then, the training image histogram is generated in two channels: *Hue* and *Saturation*, via the function *cv2.calcHist*. The next step is the probability map calculation. This operation is performed by the function *cv2.calcBackProject*, to which the input image and the histogram are given as a parameters. It is indicated, through which channels of the color space the back projection should be made. Back projection is the only way to determine how much are the pixels correspond to the histogram [43]. At the end of the preparatory stage, the function *cv2.ximgproc.createSuperpixelSLIC* initialize the object *SSllic*, and the *SSllic.iterate* method refine the boundaries.

After the above steps, we got the *PMap* and segmented image. Using them, we check the sum in each superpixel and color it out, if it does not exceed the threshold, which depends on image sizes and the number of superpixels.

The source code of the program is freely available on GitHub at https://github.com/AntonSinkov/skin-recognition.

8 Conclusion

During the research, the skin detection algorithm was developed. The influence of training images on the result is shown. The more people of different races (in approximately equal proportions), the more accurate the results of skin detection. The usage of segmentation allowed selecting skin segments, without dividing into individual pixels, which improves perception.

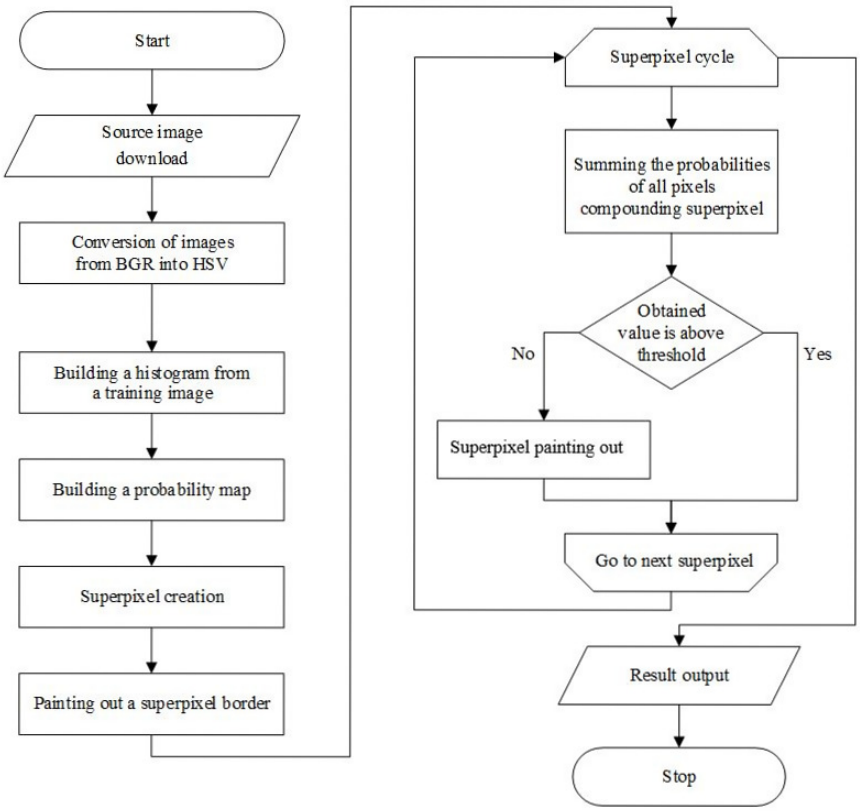

Fig. 11. Activity diagram of the developed algorithm

However, this algorithm is not absolutely accurate. Segments with a color similar to the skin color will be recognized as the skin. Nevertheless, this algorithm is suitable for preprocessing, because it is fast and accurate enough if the correct training example is provided.

In the future, it is planned to use artificial neural networks in the algorithm to speed up the information processing.

References

1. Wang X., Hu C., Yao S. An adult image recognizing algorithm based on naked body detection. In: 2009 ISECS International Colloquium on Computing, Communication, Control, and Management (ISECS '09), 8-9 August 2009, Sanya, China; 2009. p. 197–200.
2. A Comparison of Color Models for Color Face Segmentation. Procedia Technology 2013;7:134–141.
3. Albiol A., Torres .L, Delp E.J. Optimum color spaces for skin detection. In: Proceedings 2001 International Conference on Image Processing, Thessaloniki, Greece, October 7-10, 2001; 2001. p. 122–124.
4. Zhang X., Chew S.E., Xu Z., Cahill ND, SLIC superpixels for efficient graph-based dimensionality reduction of hyperspectral imagery; 2015.
5. James I.S.P. Face Image Retrieval with HSV Color Space using Clustering Techniques. The SIJ Transactions on Computer Science Engineering & its Applications 2013;1(1):17–20.
6. Caruana R., Niculescu-Mizil A. An Empirical Comparison of Supervised Learning Algorithms. In: Proceedings of the 23rd International Conference on Machine Learning ICML '06; 2006. p. 161–168.
7. Ashari A., Paryudi I., Tjoa A.M. Performance Comparison between Naïve Bayes, Decision Tree and k-Nearest Neighbor in Searching Alternative Design in an Energy Simulation Tool. International Journal of Advanced Computer Science and Applications (IJACSA) 2013;4(11):33–39.
8. Domingos P., Pazzani M. Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. In: Machine Learning Morgan Kaufmann; 1996. p. 105–112.
9. Miasnikof P., Giannakeas V., Gomes M., Aleksandrowicza L., Shestopaloff A.Y., Alam D., et al. Naive Bayes classifiers for verbal autopsies: comparison to physician-based classification for 21,000 child and adult deaths. BMC Medicine 2015;13(1):1–9.
10. Ruparel N.H., Shahane N.M., Bhamare D.P.. Learning from Small Data Set to Build Classification Model: A Survey. In: International Conference on Recent Trends in Engineering & Technology - 2013, ICRTET '2013), 15-16 March, 2013, Kodaikanal, India; 2013. p. 23–26.
11. Contributors W., Naive Bayes classifier; 2018.
12. Achanta R., Shaji A., Smith K., Lucchi A., Fua P., Susstrunk S. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. IEEE Transactions on Pattern Analysis and Machine Intelligence 2012 Nov;34(11):2274–2282.
13. Wang M., Liu X., Gao Y., Ma X., Soomro N.Q. Superpixel segmentation: A benchmark. Signal Processing: Image Communication 2017;56:28–39.
14. Manjare S., Chougule S.R. Skin Detection for Face Recognition Based on HSV Color Space. International Journal of Engineering Sciences & Research Technology 2013;2(7):1883–1887.

15. Ong P.M.B., Punzalan E.R. Comparative Analysis of RGB and HSV Color Models in Extracting Color Features of Green Dye Solutions. In: DLSU Research Congress 2014, 6-8 March, 2014, Manila, Philippines; 2014. p. 1–7.
16. Comparative Study of Skin Color Detection and Segmentation in HSV and YCbCr Color Space. Procedia Computer Science 2015;57:41–48.
17. Midha S., Vijay R., Kumari S. Analysis of RGB and YCbCr color spaces using wavelet transform. In: 2014 IEEE International Advance Computing Conference (IACC) 2014, Gurgaon, India, February 21-22, 2014; 2014. p. 1004–1007.
18. Sural S., Qian G., Pramanik S. Segmentation and histogram generation using the HSV color space for image retrieval. In: Proceedings of the 2002 International Conference on Image Processing, ICIP 2002, Rochester, New York, USA, September 22-25, 2002; 2002. p. 589–592.
19. Novak C.L., Shafer S.A. Anatomy of a color histogram. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 1992, Proceedings, 15-18 June, 1992, Champaign, Illinois, USA; 1992. p. 599–605.
20. Image Retrieval based on the Combination of Color Histogram and Color Moment. International Journal of Computer Applications 2012;58(3):27–34.
21. Han J., Ma K. Fuzzy color histogram and its use in color image retrieval. IEEE Transactions on Image Processing 2002;11(8):944–952.
22. Garg A., Roth D. Understanding Probabilistic Classifiers. In: Machine Learning: EMCL 2001, 12th European Conference on Machine Learning, Freiburg, Germany, September 5-7, 2001, Proceedings; 2001. p. 179–191.
23. Lowd D., Domingos P.M. Naive Bayes models for probability estimation. In: Machine Learning, Proceedings of the TwentySecond International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005; 2005. p. 529–536.
24. Friedman N., Geiger D., Goldszmidt M. Bayesian Network Classifiers. Machine Learning 1997 Nov;29(2-3):131–163.
25. Pal S.K., Mitra S. Multilayer perceptron, fuzzy sets, and classification. IEEE Transactions Neural Networks 1992;3(5):683– 697.
26. Chao C.F., Horng M.H. The Construction of Support Vector Machine Classifier Using the Firefly Algorithm. Computational Intelligence and Neuroscience 2015 Jan;2015:2:2–2:2.
27. Fradkina D., Muchnik I. Support Vector Machines for Classification. DIMACS Series in Discrete Mathematics and Theoretical Computer Science 2006;70:13–20.
28. Hsu C.W., Chang C.C., Lin C.J. A Practical Guide to Support Vector Classification; 2003.
29. Kim T., Park D., Woo D., Jeong T., Min S. Multi-class Classifier-Based Adaboost Algorithm. In: Intelligent Science and Intelligent Data Engineering - Second Sino-foreign-interchange Workshop, IScIDE 2011, Xi'an, China, October 23-25, 2011, Revised Selected Papers; 2011. p. 122–127.
30. Schapire R.E. In: Schölkopf B., Luo Z., Vovk V., editors. Explaining AdaBoost Berlin, Heidelberg: Springer Berlin Heidelberg; 2013. p. 37–52.
31. Thomaz C.E., Feitosa R.Q., Veiga A. Design of Radial Basis Function Network as Classifier in Face Recognition Using Eigenfaces. In: 5th Brazilian Symposium on Neural Networks (SBRN '98), 9-11 December 1998, Belo Horizonte, Brazil; 1998. p. 118–123.
32. Narayanan V., Arora I., Bhatia A. Fast and Accurate Sentiment Classification Using an Enhanced Naive Bayes Model. In: Intelligent Data Engineering and Automated Learning - IDEAL 2013 - 14th International Conference, IDEAL 2013, Hefei, China, October 20-23, 2013. Proceedings; 2013. p. 194–201.

33. Phung S.L., Bouzerdoum A., Chai D., Watson A. Naive bayes face/nonface classifier: a study of preprocessing and feature extraction techniques. In: Proceedings of the 2004 International Conference on Image Processing, ICIP 2004, Singapore, October 24-27, 2004; 2004. p. 1385–1388.
34. Park D.C. Image Classification Using Naïve Bayes Classifier. International Journal of Computer Science and Electronics Engineering (IJCSEE) 2016;4(3):135–139.
35. Sebe N., Cohen I., Huang T.S., Gevers T. Skin Detection: A Bayesian Network Approach. In: 17th International Conference on Pattern Recognition, ICPR 2004, Cambridge, UK, August 23-26, 2004; 2004. p. 903–906.
36. Azzopardi G., Petkov N., editors. The 16th International Conference, CAIP 2015, Valletta, Malta, September 2-4, 2015 Image Processing, Computer Vision, Pattern Recognition, and Graphics, Springer International Publishing; 2015.
37. Neubert P., Protzel P. Superpixel benchmark and comparison. In: Proceedings of the Forum Bildverarbeitung, January 2012, Regensburg, Germany; 2012. p. 502–513.
38. Radhakrishna Achanta KSALPF Appu Shaji, Susstrunk S. SLIC Superpixels. EPFL Technical Report 149300 2010.
39. Idris I., editor. NumPy: Beginner's Guide - Third Edition. Packt Publishing; 2015.
40. Mordvintsev A., AR K., Introduction to OpenCV-Python Tutorials; 2016.
41. Droettboom M., Caswell T.A., Hunter J., Firing E., Nielsen J.H., Varoquaux N., et al., matplotlib/matplotlib v2.0.2; 2017.
42. Mordvintsev A., AR K., Image file reading and writing; 2016.
43. Mordvintsev A., AR K., Back Projection; 2018.