TEST Engineering & Management

May - June 2020
ISSN: 0193-4120 Page No. 4382-4386

Localization of Eyes Using Haar Cascade Classifiers

[1]Aditya Kumar, [2]Ananya Shankar, [3]Akshay Ravindran, [4]Sagar K, [5]Surendra Babu K. N.

[5]Professor, [1,2,3,4,5]School of Computing and Information Technology
REVA University, Bengaluru, India
[1]adlokib@gmail.com, [2]ananyashankar.7@gmail.com, [3]akshayravindran007@gmail.com,
[4]sagarkishore.7@gmail.com, [5]surendrababukn@reva.edu.in

Article Info
Volume 83
Page Number: 4382-4386
Publication Issue:
May - June 2020

Article History
Article Received:*19 November 2019*
Revised: *27 January 2020*
Accepted: *24 February 2020*
Publication: *12 May 2020*

Abstract

When it comes to image processing, a system or an algorithm designed to process an image cannot work with an image in its raw format. Image pre-processing plays an important role in transforming or encoding an image in a form that can be comprehended by the algorithm/system. This may include extraction of certain features or combining features to transform the image into a desirable format. One of the most essential pre-processing tasks is the localization of objects/areas of interest in an image. Localization focuses on detecting only the parts of the image that are to be processed while ignoring the rest of the image. This paper describes an effective method in localizing the eye region from an image which contains the upper region of a human face. The dataset used to perform localization of the eye region is the second subset of the CASIA-Iris-Mobile-V1.0. It is done using a combination of various Haar Cascade classifiers.

Keywords: *Localization of Eyes, Digital Image Processing, Image Pre-processing.*

1. Introduction

Localization is a pre-processing technique which is used to detect/extract only the elements of an image which are necessary for further processing. It is easier to work with an image which contains only the necessary information with minimal noise. Anything that is not required to be processed or considered useless can be removed to make the processing conducted in later stages more efficient and robust. This paper discusses about how the human eye region from an image of the face can be detected and localized for further processing. The technique used in this paper can be divided into 4 modules; namely, image transformation, conditional classifier selection, prediction agglomeration and transitive extraction.

Every image in the dataset contains images of the face with both the eyes. The size of the original image from which the eye region has to be localized is 1968 x 1024 pixels. In the first module, which is image transformation, the original image is reduced to 10 percent of its original size, i.e., 196 x 102 pixels.

In this paper, we have used a combination of multiple haar cascade classifiers to localize both the left and right eye regions from the image. Based on certain conditions specified, the transformed image is fed into one or more classifiers making the technique more robust. This is carried out in the conditional classifier selection module. Since the transformed image has been fed into one or more classifiers, there will be one or more predictions of the eye region. In the prediction agglomeration module, the multiple predictions are agglomerated to produce a single eye region which is more accurate.

The final prediction in the previous module is made on the transformed image. This prediction is applied to the original image in an appropriate fashion in the transitive extraction module. Finally, the predicted eye region is cropped out from the original image.

The decision to use haar cascade classifier was made due to its ability to handle images of any resolution. This makes the classifier capable of localizing the eye region in images with various resolutions. We also have the added advantage of providing real-time localization and

May - June 2020
ISSN: 0193-4120 Page No. 4382-4386

are extremely computationally efficient. They work with barely 50 percent of the image data but are capable of predicting 99 percent of the targeted output. They operate under the concept of sub-windows and resolution pyramids. Any haar cascade contains a certain number of classifiers which gets progressively more complicated as the earlier ones are used to detect low level features and the later ones more complex features. Thereby, if a sub-window is rejected by an earlier classifier, it is not processed by the subsequent ones. Using several classifiers in combination provides us with extremely robust predictions of the eye region which are capable of handling nearly any input.

2. Literature Survey

Eye localization is the first part of the iris verification pipeline which is considered one of the most vital pre-processing steps for increasing the efficiency of the system. And as such, there has been a lot of research work carried out in the field of ocular biometrics [5]. Eye localization can be considered as the most important step in the pipeline as the subsequent modules depend upon it to provide accurate input for further processing. If this module fails to perform efficiently enough, there will be a cascading effect on the performance of the subsequent modules. There is much work carried out in this field, the most notable of which is by Daugman et al [20]. Daugman's method is one of the first methods to perform iris verification accurately and boasts a false acceptance rate of one in 4 million, and has proved to be the definitive algorithm to be used in terms of iris verification to this day. Hough transform is another algorithm which is often used when performing the iris segmentation task [7]. The requirement of this particular iris segmentation algorithm as well as other approaches such as U-net Deep Convolutional Neural Network models [1] is that we provide a properly localized eye to it as an input. In recent times, deep neural networks have proven to be much more adept at iris segmentation as they are not limited to only extracting an annulus iris region by finding two circles in the image like Daugman's algorithm or the Hough transform approach. They can be trained to find asymmetric iris regions containing eyelids, eyelashes and other artefacts [4]. The methodology laid out in [2] provides the outline necessary for carrying out the iris verification process using more than one facial features. In it, eye localizations have been carried out using the AdaBoost classifier [18]. Furthermore, [8] outlines a hard-computing methodology for iris segmentation. Regardless of whether the iris segmentation algorithm is a soft or hard algorithm, both require an accurately localized eye for further processing in order to obtain better results. For obtaining an appropriate feature representation of the segmented iris, the one mentioned in [6] proves to be the most efficient method. Finally, proper eye localization would be crucial in mission critical systems such as secure authentication systems [3].

3. Technical details

A. Dataset

The dataset utilized is the publicly available CASIA-Iris-Mobile-V1.0-S2 (Second subset). The dataset consists of a total of 6000 images. These images were collected from a total of 200 subjects. 10 images of each subject were captured from 3 different distances giving a total of 30 images per subject. The distances at which the subjects' images were captured are 20, 25 and 30 centimeters. The images were captured using an external NIR setup which consisted of multiple NIR LEDs surrounding the camera. The captured images have a resolution of 1968 x 1024 pixels and are in grayscale. The dataset primarily consists of subjects of Asian ethnicity. Each image consists of the upper face region, specifically, the eyes and the bridge of the nose. Some images also contain other artefacts such as hair and eyebrows. Occasionally, few images contain a partial background as well.

B. OpenCV

The localization algorithm mainly uses the OpenCV Python library. OpenCV provides a wide variety of functions pertaining to Computer Vision. Some of the functions we have utilized include cv2.cvtColor(), cv2.resize(), cv2.imread() and cv2.imwrite(). When resizing the image, the interpolation used is cv2.INTER_AREA. OpenCV provides numerous pre-trained
haar cascade classifiers for detecting various artefacts. Of these, 4 classifiers are provided for the purpose of locating eyes.
These include haarcascade_eye.xml,
haarcascade_eye_tree_eyeglasses.xml,
haarcascade_lefteye_2splits.xml
and haarcascade_righteye_2splits.xml.
All the classifiers are trained using a 20 x 20 pixels window. The first classifier is a stump-based frontal eye detector. The second classifier is a tree-based frontal eye detector with better handling of eyeglasses. The third and fourth classifiers are also tree-based. The third classifier is a left eye detector while the fourth one is a right eye detector. Both these detectors are trained by 6665 positive samples from FERET, VALID and BioID face databases.

C. Environment and Optimization

We have used Python language with Jupyter Notebook and the entire setup has been configured in a conda environment. The libraries used include numpy, h5py, opencv, glob and matplotlib for visualization.

In order to reduce the load time of the entire dataset containing 6000 images, the dataset was initially resized 1 image at a time, converted into a numpy array and saved as an h5 file resulting in a much better time complexity. Where it took several minutes to load all the 6000 images into the Python environment using cv2.imread(), it only takes few seconds to load the entire dataset using h5py.

May - June 2020
ISSN: 0193-4120 Page No. 4382-4386

4. Methodology

The localization algorithm used in this paper is sub-divided into 4 parts. They are:

- Image Transformation
- Conditional Classifier Selection
- Prediction Agglomeration
- Transitive Extraction

A. Image Transformation

As described earlier, the original images from the dataset are resized in such a manner that the new image has a resolution that is 10 percent of the original resolution; thereby having a size of 196 x 102 pixels. The reason the images are resized instead of using the uncompressed raw images is that it provides us an exponentially better space and time complexity while still retaining all the necessary information contained in the original image required for localizing the eyes. Once the images are resized, all 6000 of them are made into a single numpy array and saved as a h5 file which makes it easier to access them later as needed. The haar cascade classifier has been trained with a window of 20 x 20 pixels and therefore can much more easily localize the eyes in the resized image.

B. Conditional Classifier Selection

The algorithm consists of global parameters which include the width of the resized image, the mid-point of the width, a variable p whose value has been found to be 5.16, another variable q whose value has been found to be 3.3 and a variable j whose value has been found to be 5.0. p, q and j have been found through trial and error as they provide the most optimal predictions when used with haar cascade classifiers. The variable mins signifies the minimum width of any prediction made by the classifiers and its value is set to be width/p. The variable bigs signifies the maximum width of any prediction made by the classifiers and its value is set to be width/q. The last global parameter is called size and it represents the width of the final crop which will be extracted from the original image.

For each image, there exists few local parameters which are reset whenever a new image is to be processed. These include flagL which is set to 0 representing that the left eye has not been found yet in the image and flagR which is also set to 0 representing that the right eye has not been found yet in the image. Subsequently, whenever the left or right eye is localized, these flags are set to 1. Leye and Reye are initialized as empty lists and will contain the predictions for the left and right eyes when the classifiers are used.

Each time a classifier is utilized, it is passed 4 parameters. The first being the grayscale image itself, the second being the scale factor, the third being minSize and lastly maxSize. The most important parameter of these 4 is the scale factor which has been set to a value of 1.005. We arrived at this value through trial and error.

For haar cascade classifiers, the scale factor determines how fine the scaling pyramid is going to be. Its value should always be greater than 1. The scaling pyramid is created by taking the image and reducing its resolution by the scale factor to create the next level. A greater scale factor results in fewer stacks whereas a smaller scale factor results in a larger number of stacks. If we use a larger scale factor, the execution time of the classifier reduces but there is a possibility of missing out certain artefacts. Whereas, using a very small scale-factor causes the execution time to increase and can produce many false positives. Therefore, it is necessary to finetune the scale factor to our needs appropriately.

In those conditions where the left or right eye classifier is used, the predictions have an anomaly of having the eye in the lower portion of the bounding box. Therefore, the position of the bounding box is adjusted in such a way that the eye is roughly in the center of the bounding box. The distance by which the bounding box is moved is equal to the width of the bounding box divided by the variable j.

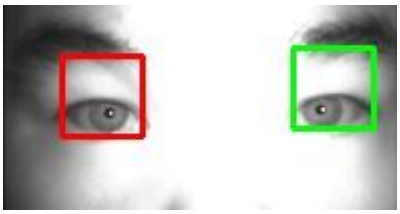

Figure 1: (a) Bounding box positioning anomaly.

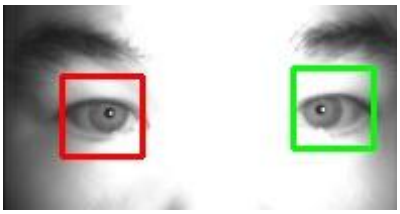

Figure 1: (b) Corrected position of bounding box.

Once the list of eyes is obtained from the haar cascade classifiers, they are segregated into Leye or Reye. If the midpoint of a predicted eye is less than the midpoint of the image, it is classified as a left eye. Whereas, if the midpoint of a predicted eye is greater than the midpoint of the image, it is classified as a right eye. Whenever an eye is classified into the left or right list, the respective flag is set to 1.

C. Prediction Agglomeration

Once the transformed image passes through the various haar cascade classifiers, we obtain a list of bounding boxes for each eye that contains one or more bounding box. Now we take the list for each eye and find the mean values of their x-coordinate, y-coordinate and width. This gives us one bounding box for each eye.

May - June 2020
ISSN: 0193-4120 Page No. 4382-4386

D. Transitive extraction

Once we've obtained the bounding box for an eye, we can proceed to use it to extract the eye localization from the original image. The bounding box is described by a list of 3 parameters namely, x-coordinate, y-coordinate and width. The x and y coordinates correspond to the top left corner of the bounding box and the width is the width of the bounding box which is the same as its height. Since we had reduced the size of the original image to 10 percent of the original size, we can seamlessly obtain the coordinates and the width of the bounding box for localizing the eyes in the original image simply by multiplying the 3 parameters which we had obtained earlier by 10. For example, if the 3 parameters in the resized image are 23, 16 and 48, that is, the coordinates are (23,16) and the width is 48 then the parameters of the bounding box in the original image will simply be 230, 160 and 480 respectively.

Once we have these 3 parameters, we calculate the coordinates of the center of the bounding box by adding half the width to the x and y coordinates. From the center point, we calculate the final x and y coordinates by subtracting half the value of the variable named size. For instance, if the center of the bounding box is (302,410) and the size of our bounding box is supposed to be 400 pixels, the final x and y coordinates are 302 - (400 / 2) = 102 and 410 - (400 / 2) = 210, thus giving us (102,210) as the x and y coordinates of the bounding box in the original image and 400 pixels as the width of the bounding box. We do this so that each eye localization that we obtain has a fixed width (We have chosen 400 pixels as the standard width). Using these 3 parameters, we crop the localized eye from the original image. This entire process is carried out for both the left and right eyes.

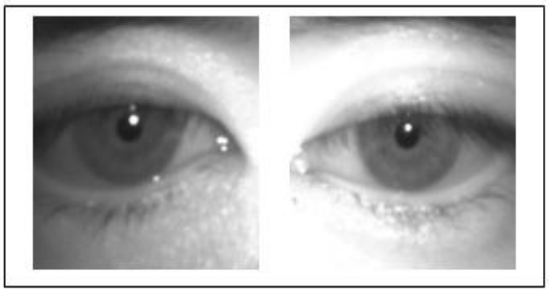

Figure 2: (a) Cropped left eye.

(b) Cropped right eye.

5. Conclusion

In this paper we have outlined an eye localization algorithm which uses multiple pre-trained haar cascade classifiers available through open software libraries and have applied it on the CASIA-Iris-Mobile-V1.0-S2 (Second subset).

This localization algorithm proves to be extremely robust and provides an accuracy of 99.75%.

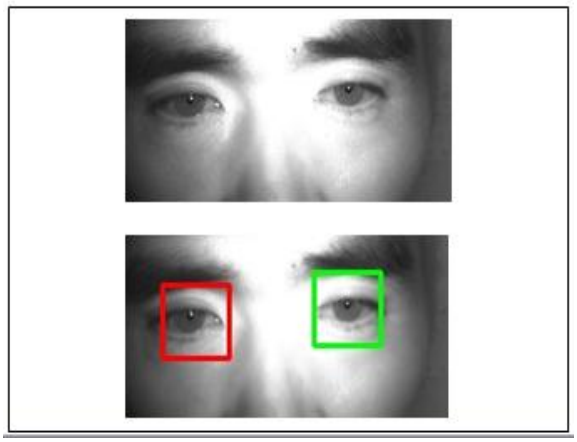

Figure 3: (a) Original image with upper face region.

(b) Image with localized eyes.

Finally, to sum up, the model is sufficiently capable of localizing the eye from an image containing the upper part of a human face.

6. Scope and Future Enhancement

Localization is a vital pre-processing technique which reduces computational complexity in computer vision tasks. Localization of eyes is the first step in the iris recognition pipeline which helps in making the processing in later stages easier and faster. In the future, we plan to incorporate an iris segmentation module to extract only the iris region of the localized eye using a deep convolutional neural network and a feature vector generation algorithm thereby creating a complete iris verification system which can be used for secure authentication in various settings.

References

[1] Lozej, Juš, et al. "End-to-end iris segmentation using U-Net." 2018 IEEE International Work Conference on Bioinspired Intelligence (IWOBI). IEEE, 2018.

[2] Zhang, Qi, et al. "Fusion of face and iris biometrics on mobile devices using near-infrared images." Chinese Conference on Biometric Recognition. Springer, Cham, 2015.

[3] Albadarneh, Aalaa, Israa Albadarneh, and Ja'far Alqatawna. "Iris recognition system for secure authentication based on texture and shape features." 2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT). IEEE, 2015.

[4] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015.

[5] Nigam, Ishan, Mayank Vatsa, and Richa Singh. "Ocular biometrics: A survey of modalities and

May - June 2020
ISSN: 0193-4120 Page No. 4382-4386

fusion approaches." Information Fusion 26 (2015): 1-35.

[6] Wang, Libin, Zhenan Sun, and Tieniu Tan. "Robust regularized feature selection for iris recognition via linear programming." Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012). IEEE, 2012.

[7] Koh, Jaehan, Venu Govindaraju, and Vipin Chaudhary. "A robust iris localization method using an active contour model and hough transform." 2010 20th International Conference on Pattern Recognition. IEEE, 2010.

[8] He, Zhaofeng, et al. "Toward accurate and fast iris segmentation for iris biometrics." IEEE transactions on pattern analysis and machine intelligence 31.9 (2008): 1670-1684.

[9] Schmidt, Adam, and Andrzej Kasiński. "The performance of the haar cascade classifiers applied to the face and eyes detection." Computer Recognition Systems 2. Springer, Berlin, Heidelberg, 2007. 816-823.

[10] Kasinski, Andrzej, and Adam Schmidt. "The architecture of the face and eyes detection system based on cascade classifiers." Computer Recognition Systems 2. Springer, Berlin, Heidelberg, 2007. 124-131.

[11] Campadelli, Paola, Raffaella Lanzarotti, and Giuseppe Lipori. "Eye localization: a survey." NATO SECURITY THROUGH SCIENCE SERIES E HUMAN AND SOCIETAL DYNAMICS 18 (2007): 234.

[12] Everingham, Mark, and Andrew Zisserman. "Regression and classification approaches to eye localization in face images." 7th International Conference on Automatic Face and Gesture Recognition (FGR06). IEEE, 2006.

[13] Dobeš, Mihal, et al. "Human eye localization using the modified Hough transform." Optik 117.10 (2006): 468-473.

[14] Niu, Zhiheng, et al. "2d cascaded adaboost for eye localization." 18th International Conference on Pattern Recognition (ICPR'06). Vol. 2. IEEE, 2006.

[15] Campadelli, Paola, Raffaella Lanzarotti, and Giuseppe Lipori. "Precise Eye Localization through a General-to-specific Model Definition." BMVC. Vol. 1. 2006.

[16] Wang, Qiong, and Jingyu Yang. "Eye detection in facial images with unconstrained background." Journal of Pattern Recognition Research 1.1 (2006): 55-62.

[17] Wilson, Phillip Ian, and John Fernandez. "Facial feature detection using Haar classifiers." Journal of Computing Sciences in Colleges 21.4 (2006): 127-133.

[18] Viola, Paul, and Michael J. Jones. "Robust real-time face detection." International journal of computer vision 57.2 (2004): 137-154.

[19] Feris, Rogerio Schmidt, et al. "Hierarchical wavelet networks for facial feature localization." Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition. IEEE, 2002.

[20] Daugman, John G. "High confidence visual recognition of persons by a test of statistical independence." IEEE transactions on pattern analysis and machine intelligence 15.11 (1993): 1148-1161.