Access Control in the Wild Using Face Verification

Ricardo Ribeiro, Daniel Lopes and António Neves

Additional information is available at the end of the chapter

Abstract

In the past few years, face recognition has received great attention from both research and commercial communities. Areas such as access control using face verification are dominated by solutions developed by both the government and the industry. In this chapter, a face verification solution is presented using open-source algorithms for access control of large-scale events under unconstrained environments. From the type of camera calibration to the algorithms used for face detection and recognition, every stage has a proposed solution. Tests using the proposed solutions in the entrance of a building were made in order to test and compare each solution proposed.

Keywords: face recognition, face detection, access control, unconstrained environment, camera calibration

1. Introduction

Over the past few years, face recognition has become one of the most successful applications in computer vision and pattern recognition. It has received significant attention in several areas, such as law enforcement and surveillance (video surveillance and access control), smart cards (national ID and passports), information security (data management and file encryption), and entertainment (video game and virtual reality), among others [1].

Biometric-based technologies have been developed and implemented over the last century. These systems are the most promising for personal identification. Examples of modes of biometric systems include face recognition, fingerprints, iris scanning, and others [1].

Face recognition as a biometric technique appears to offer several advantages. The lack of interaction of the user is an important advantage regarding these types of systems appointed by [1]. In a fingerprint system, for example, the user needs to place his finger in a designated area, while in a face recognition system, the face images can be acquired passively.

Face recognition systems have, however, some level of complexity as there are some stages that are needed to execute in order to achieve a system with a good performance. **Figure 1** presents these stages.

Within each stage, there are specific operations that can be added in order to achieve better performance results. Right on the start, the image acquisition is a crucial step where there is room for improvement. Later, the face detection and recognition can be performed by specific algorithms which are presented and studied. Finally, two algorithms for face normalization (also known as preprocessing) algorithms, which are mentioned on state of the art articles, are also analyzed for this specific chapter.

State of the art face recognition is dominated by industry- and government-scale datasets. There is a large accuracy gap between today's publicly available face recognition systems and the state of the art private face recognition systems [2]. However, this gap is closing up as better open-source algorithms and datasets with more and better images starts to appear.

Despite the success and high verification or recognition rates, there are still some challenges such as age, illumination, and pose variations. Most of these systems work well under constrained conditions (i.e., scenarios in which at least a few of the factors contributing to the variability between face images are controlled); however, the performance degrades rapidly when they are exposed to conditions where none of these factors are regulated [3].

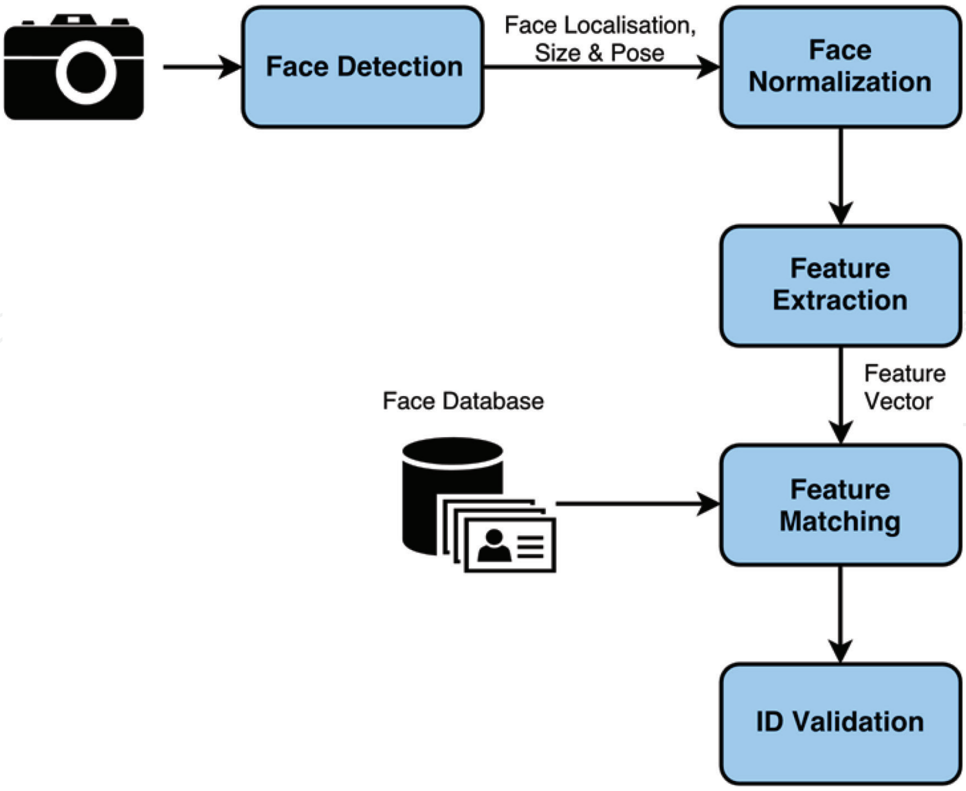

Figure 1. Configuration of a generic face recognition system.

Toward exploring this field and the increasing demand of these systems, an access control solution for unconstrained environments using face verification with open-source algorithms is presented in this chapter.

An introductory section is presented that provides a brief introduction to the face recognition system. In Section 2, the proposed solution is described. Later in this section, the major problems for a face recognition system for unconstrained environments are explained. These problems are some of the challenges that are tried to solve in this chapter. The several implemented algorithms are described in Section 3. In Section 4, experimental results showing the effectiveness of the proposed algorithms and the comparison between them are provided. Finally, a summary of the work done, comparison of the different experiments, concluding remarks, and the future work are featured in Section 5.

2. Proposed solution

The proposed solution consists of the creation of a face verification (1:1 match comparison) system using open-source face recognition and detection algorithms in order to implement it in large-scale events with access control, such as sport infrastructures. To access this type of events, it is usually through the acquisition of a ticket/ID card. In order to improve the access control, the ticket access/acquisition is complemented with a face verification system.

The environment of these places is usually outdoors; therefore, the lighting conditions cannot be fully controlled [4]. Thus, the solution involves the use of cameras with adjustable parameters which do not have a proper calibration for these types of environments. An artificial light is also added which helps to compensate the lack or the excess of light in the scene.

As for software, two different programming languages, C++ and Python, are used. The C++ language is used for image acquisition and control of the camera parameters, including the calibration method since the cameras manufacturer only provides a library for the C programming language. Once the image is acquired, it is sent through a socket to a Python script that uses all the computer vision algorithms.

The solution is divided into two stages, registration and verification, that work independently of each other. Each process is divided into five scripts, where camera, facetracker, and NFC scripts are common to both stages. The database and interface scripts have some differences in both stages.

The different scripts are described as follows:

1. *Camera Script*: This script developed in C++ starts the system with the acquisition of images from the outside world. When a person approaches and his/her face is detected, a calibration of the camera parameters is done. Once the camera calibration is finished, the images acquired are sent in real time to the facetracker script.
2. *Facetracker Script*: Once activated, this Python script tracks the face of the person who is in front of the camera. The face is analyzed in order to crop and process only persons looking

toward the camera. When the face cropping and the preprocessing is done, the face image goes through a Deep Neural Networks (DNNs) which gives a 128D vector as output. This vector is compared to the vector acquired on the previous face image and, if the threshold is above the limit set, it will mean that a different person who is appearing in front of the camera, thus sending a warning to the next script. When the comparisons are below the threshold, the output vectors of the DNN are sent to the script.

3. *Database Script*: If no warning is received from the previous script, this script will store the output vectors of the face images. For the verification stage, when it receives an ID from the NFC script, these vectors are compared to the ones that are in the database associated with the ID number. The comparison gives, like in the case of the facetracker, an output value that decides if the person has that card associated with him/her. According to the comparison value, a token is sent to the Interface script. For the registration, instead of comparing the vectors, they are stored in the database associated to the ID number.
4. *NFC Script*: Python script that reads the NFC card reader values. If someone swipes an NFC tag over the card reader, the ID value of that tag is sent to the Database Script.
5. *Interface Script*: This script is used for the communication between the user and the system. It is designed to show images on the screen that gives feedback to the user of what is happening in the system. It tells the user to look at the camera, if he/she had done the registration, and finally a welcome message is displayed if the comparison value presented on the database script is below the threshold or a denying message if the value is above. When the person is in the registration, a wait message is displayed while the vectors are stored and finally it shows a successful registration message.

Figure 2 shows a block diagram of scripts communication.

2.1. Challenges

In order to build a face verification system with these characteristics, an important factor is taken into account: the unconstrained environment where the system is going to be implemented. In a computer vision point of view, some problems related with these kinds of environments appear which are mentioned as follows:

1. ***Head pose:*** At the time of the image acquisition, the viewing direction of the subject may not be toward the camera. These face images may not be the best suitable for the face recognition system.
2. ***Face image resolution:*** As the subject approaches the camera, his/her face starts to be detected. However, if the person is still at some distance from the camera, the face images collected may not have enough resolution for the system.
3. ***Subject motion:*** It is taken into consideration that the subject is in movement and that may cause some blur in the images acquired.
4. ***Face tracking:*** It is crucial that there will be distinction between different subjects especially at the time of the ticket acquisition as if not done correctly, the face images of different subjects may end up in the same person database.

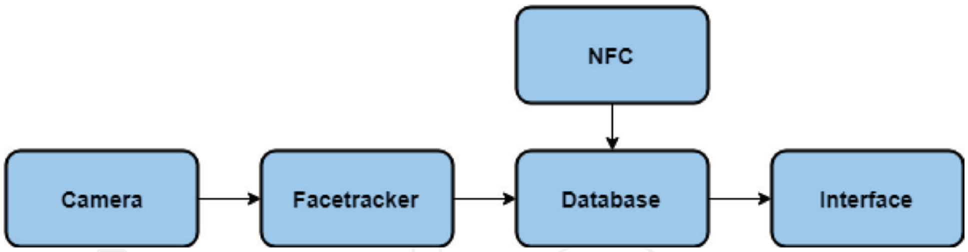

Figure 2. Block diagram of communication between scripts.

5. ***Non-controlled illumination:*** This may be the most difficult challenge to overcome as the cameras may be installed in an outdoor environment and, therefore, different lightning conditions according to the time of the day and the meteorological circumstances.

3. Proposed algorithms

The software developed obeys to some specific steps which are exposed in **Figure 3**.

3.1. Calibration through intensity pixels

In this section, a different type of calibration is proposed to acquire the best digital image for the face verification system.

When using the automatic calibration of the parameters provided by the camera, the whole image is considered when calibrating. Therefore, the region of interest (ROI) that will be acquired by the system can be affected by the light intensity that there is in the background and the image may not have the best quality.

In order to get the most suitable ROI (in this case, the face) for the system, it is attempted to create a calibration focused on this region.

The algorithm proposed is a mixture between the calibration of exposure time and gain.

Since the main goal is to implement the face verification system in an uncontrolled environment, an initial calibration is done using the auto-parameters calibration provided by the camera in order to adapt to the light and environment conditions and to detect the first face for the

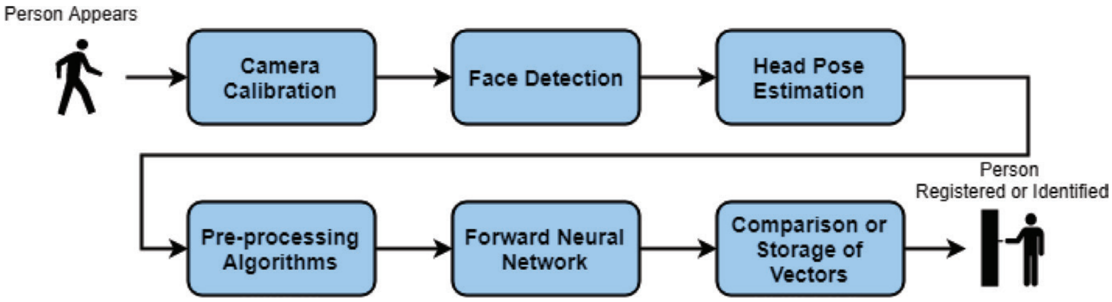
Person Appears
Camera Calibration
Face Detection
Head Pose Estimation
Pre-processing Algorithms
Forward Neural Network
Comparison or Storage of Vectors
Person Registered or Identified

Figure 3. Workflow diagram of the software developed.

use of the calibration. At this point, a timer is set to wait a few seconds, so that the parameters of the camera have the time to be internally changed and established. Exposure time, gain, and white balance are the parameters changed automatically by the camera software.

When a face is found, the auto-parameter calibration is disabled and it continues to the next step of the calibration.

3.1.1. Mean sample value

This calibration step is based in the mean sample value (MSV) from the image gray-level histogram of the region where the face is represented. Introduced by [5], the MSV is used to calibrate the exposure time and the gain of the camera.

In this stage, the MSV is calculated through the gray-level histogram of the face region with the equation described next:

$$\mathrm{MSV} = \frac{\sum_{j=0}^{4} (j+1)\, x_j}{\sum_{j=0}^{4} x_j} \tag{1}$$

where x_j is the sum of the gray values in region j of the histogram (in the proposed approach, the histogram is divided into five regions).

It is worth noting that the pixels used for the MSV calculation are only the ones that are inside of the face-bounding box of the largest face found on the image.

A range of values is set for the MSV. If the calculated MSV is in range, between the values 2.1 and 2.5, the camera parameters (gain and exposure) have acquired values. Otherwise, the camera parameters are increased or decreased depending on whether the MSV is below the minimum value or above the maximum value, respectively.

This method has the main advantage that, if the same person appears on different parts of the day, the face images acquired will have very similar intensity values as the gain is calculated to have the same intensity values between a certain range.

3.1.2. White pixels

This method addresses the situations when the face of a subject is partially exposed to sunlight which causes that part of the face too bright. To solve this, if a region where the intensity pixels have the maximum intensity found, the camera parameter values are decreased in order to reduce the brightness of that region of the face. In this case, when reducing the camera parameters, the side of the face that is not exposed to sunlight may become too dark. Therefore, the MSV value cannot be reduced far below the minimum value previously.

Figure 4 shows the comparison between parameter calibration provided by the camera and the proposed calibration, respectively.

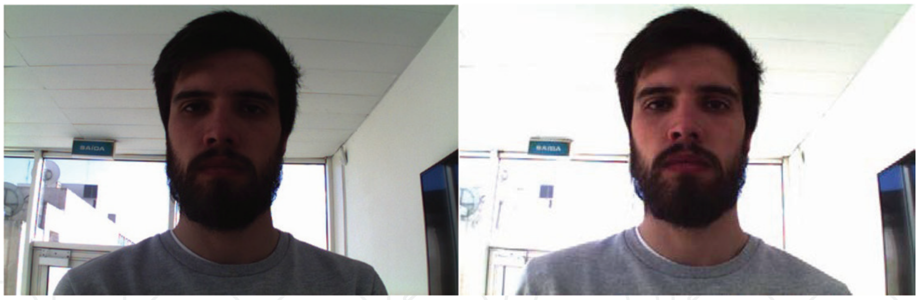

Figure 4. Comparison between the automatic calibration (left image) and the proposed calibration (right image).

3.2. Detection and recognition algorithms

Several algorithms were studied and implemented into the system. Despite these algorithms being state of the art, where the use of neural networks is prevalent in an attempt of closing the gap between the performance of commercial and open-source of face recognition solutions, several other exist in the study [6].

In previous work [7], the Haar Cascades, Local Binary Patterns Cascade (LBP), and Histogram of Oriented Gradients (HoG) algorithms were studied for face detection. The FisherFaces and Local Binary Patterns Histograms (LBPH) algorithms were also studied for face recognition.

3.2.1. Face detection

1. ***Histogram of oriented gradients (HoG)***: *Dlib's*[1] implementation is based on the algorithm presented in [8] that it is used for the face detection stage. It is especially useful as it provides 68 face landmarks that are further used at the recognition step for pose estimation.
2. ***Multi-task cascaded convolutional networks*** [9]: Deep cascaded multi-task framework exploits the inherent correlation between detection and alignment to boost up their performance. It provides five major face landmarks instead of the 64 of *Dlib*. It is, however, more immune to light variations and occlusion.

3.2.2. Face recognition

1. ***Deep metric learning (DML):*** Implementation also provided by *Dlib* library where the network implemented was inspired in [10] that does the face verification. The model trained achieves 99.38% in the benchmark Label Faces in the Wild (LFW) [11]. The input data of the network model for training were two datasets: the FaceScrub dataset [12] and the VGG dataset [13] with about 3 million faces in total.
2. ***OpenFace*** [2]**:** Face recognition with deep neural networks which achieves an accuracy of about 92% on the LFW [11] benchmark. The training of the neural network was done with the CASIA-WebFace [14] and FaceScrub [12] containing about 500,000 images.

[1] http://dlib.net/

3. ***DeepFace*** [13]**:** Algorithm inspired in [15, 16]. The CASIA-WebFace is used on training. In LFW benchmark, it achieves 99.2% of accuracy. The implementation used of this algorithm can be found in the github repository.[2]

It is worth mentioning that the *OpenCV*[3] library was used in the image processing and transformation.

3.3. Preprocessing methods

Once the image is acquired, there is still some image processing that may improve the system accuracy toward detection and recognition.

3.3.1. Gamma correction (GC)

Gamma is a very important characteristic in any digital system. In the world of cameras, it defines the relationship between a numerical value of a pixel and its actual luminance. The GC enhances the local dynamic range of the image in dark or shadowed regions while compressing it in bright regions and at highlights [17]. However, this operation is still affected by some level of directional lightning as pointed by [18].

Given a certain gamma (γ), the relation between the gray-level image with gamma correction (I_g) and the original one (I) is given by $I_g = I^{\gamma}$.

As it is possible to analyze in **Figure 5**, the human eye does not relate the detected light with the actual luminance as a "linear" relationship.

Figure 6 presents images with different gamma values, from the highest value to the lowest value (from the left to the right). As it is possible to analyze, the image with a higher gamma is more uniform regarding light.

The ambition then is that using an appropriate gamma value, the images acquired will not be as susceptible to lighting variations.

3.3.2. Contrast-limited adaptive histogram equalization (CLAHE)

CLAHE is an adaption of Adaptive Histogram Equalization (AHE) [19] that was first introduced for contrast enhancement for both natural and non-visual images [20]. This variation that introduced the limitation of contrast started began to be used in the face recognition field [21], which improved the contrast in face images.

Later, it began to see its utility in the facial recognition field, and a variation entitled contrast-limited adaptive histogram equalization (CLAHE) [19] was started to be used.

CLAHE is a preprocessing stage that focuses on improving the contrast in an image. This technique was applied as a preprocessing technique in [21] and it was applied on the face

[2] https://github.com/bearsprogrammer/real-time-deep-face-recognition

[3] https://opencv.org/

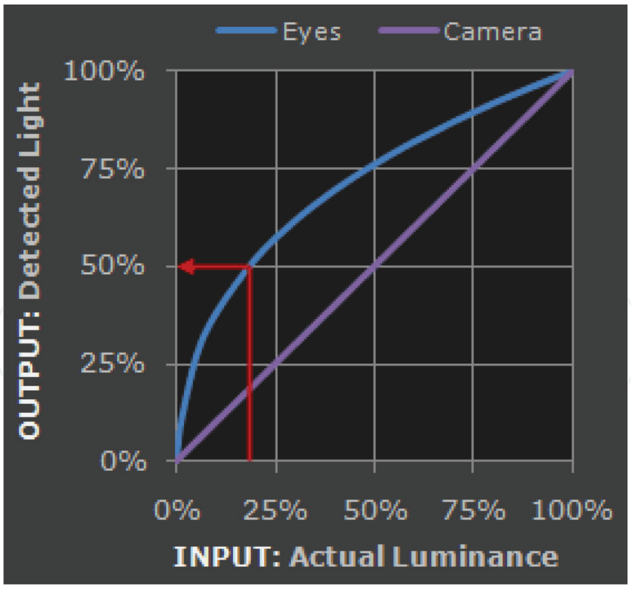

Figure 5. Relation between the detected light and the actual luminance for both eyes (in blue) and camera (in purple) (http://www.cambridgeincolour.com/tutorials/gamma-correction.html).

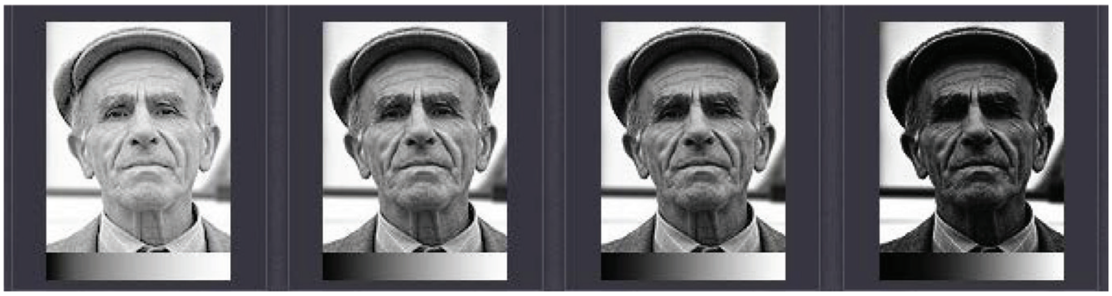

Figure 6. Face images with different gamma values. From the highest value to the lowest value (from the left to the right) (http://www.cambridgeincolour.com/tutorials/gamma-correction.html).

images in order to highlight the features that describe the face. The results exposed regarding recognition were improved with the addition of this stage.

In this approach, the face image is divided into small blocks, also called tiles, and in each of these blocks, the histogram equalization is applied. However, if any of the histograms calculated is above the predefined contrast limit, the pixels are clipped and distributed uniformly to other bins before applying histogram equalization. **Figure 7** shows a face image before and after the application of the CLAHE, respectively.

3.4. Head pose estimation

In previous work, three degrees of freedom were used for face alignment [7]: yaw, pitch, and roll which are presented in **Figure 8**.

In order to filter some of the input faces in face recognition algorithms, we use these three values to estimate the face position. For each value, a range is defined so that only the faces that are almost frontal to the camera are accepted as input of the face recognition algorithms.

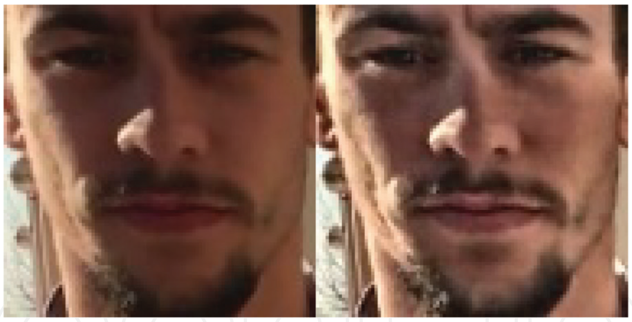

Figure 7. Face image without (left image) and with (right image) the application of CLAHE.

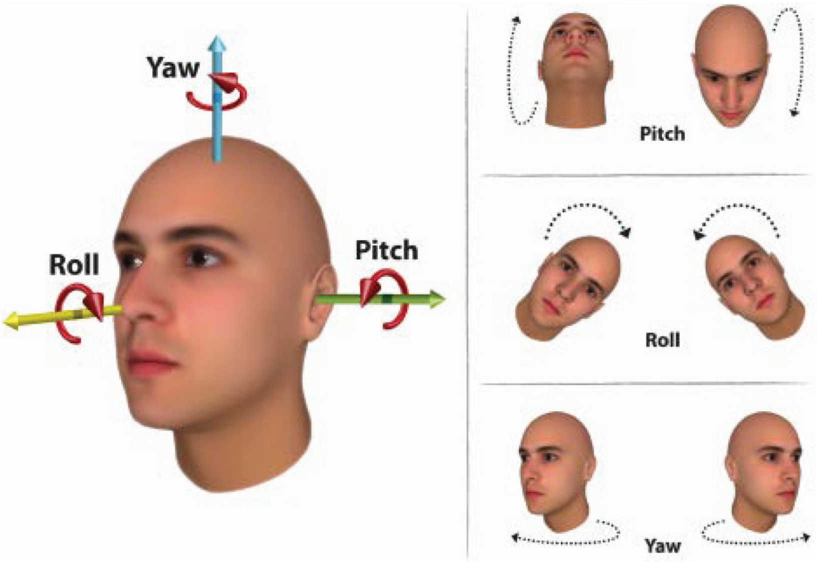

Figure 8. The three degrees of freedom of a human head that can be described by the egocentric rotation angles pitch, roll, and yaw [22].

4. Experimental results

In order to test the algorithms for the proposed solution, an access control system was simulated with face verification at the entrance of the research institute where dozens of people come and go during the day.

4.1. Setup

A prototype was developed with the available material. The prototype incorporates two cameras, an industrial camera and a webcam, on a tripod placed at a height of 1.5 m. Above the camera, an artificial illumination was placed. Since there were no available processing boards during the development of the system, a personal computer (ASUS VivoBook S14), Intel Core i7 8550U, was used instead. Finally, an NFC card reader (RFID-RC522) was connected to read the tag ID of each user both on the registration and verification stages. Also, it is worth mentioning that the PC display was used to show interactive messages explaining what the user should do. **Figure 9** presents the setup of the system.

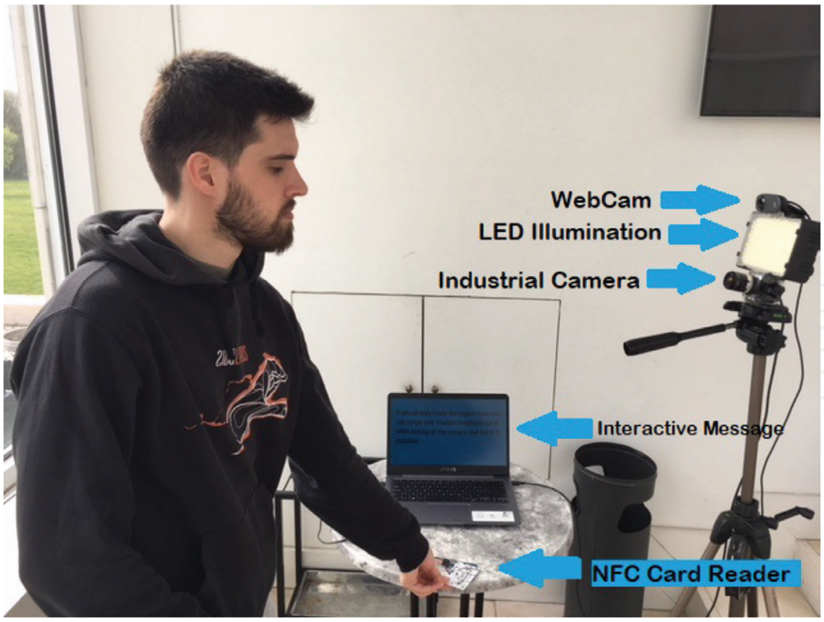

Figure 9. User interacting with the system setup for the experimental results.

4.1.1. Cameras

The cameras used for tests are the IDS UI-1220LE-C (Industrial Camera) and the Logitech C310 (webcam). The purpose of the use of these cameras is to compare the performance between them in this specific system as the webcam does not allow to change its camera parameters such as exposure or gain.

On the other hand, the industrial camera, despite not being the most suitable for this scenario, provides a software development kit (SDK) that enables the complete control of its different parameters. In addition, as the industrial camera does not have a lens integrated, a 4.5-mm lens with manually adjustable aperture is used. Since the system is intended to be implemented at a fixed location, the most suitable lens aperture is defined.

In order to access the image data directly and to process the image captured by the webcam, the Video4Linux API using the OpenCV Video I/O module is used.

As for the industrial camera, an initial procedure is required to prepare the image capture.

In the first phase, the industrial camera is initialized to establish the connection. In order to obtain the image for the system, access to the image data stored in memory is required. To do that, it is necessary to obtain the sensor size as it determines the memory needed to allocate the image.

4.1.2. Illumination

A 168 LED illumination with adjustable intensity is used in order to compensate the excess or lack of illumination. It also eliminates any occlusion that may be caused by external lightning. Another major advantage is its use on darker scenarios where the camera has a substantial exposure time. If the illumination is turned on, the scenario is clearer and the exposure time needed is lower, thus the blur caused by the person's motion in the image is less than without illumination.

4.2. Description of the experiment

The tests were done in three distinct days where the first and the third day were sunny and the second one was cloudy. People who were entering the building were asked if they want to participate in this study. If the person agreed, he/she posed himself/herself in front of the camera and the registration was done (if it was the first time that the person presented in front of the camera). As for the next times the person appeared, the comparison between the face images made on registration and the ones acquired at the time was made. **Figure 10** shows some of the face images acquired in different days.

About 50 people (a big majority of Caucasians from both sexes) participated, and all the participants entered the building at different times of the day which caused different types of directions of lighting in the face images acquired.

The comparisons between the face images registered in the database and the ones acquired next gave output values which were used to construct the receiver operating characteristic (ROC) curves. In total, about 2500 comparison values with both false and true positives were used to construct each curve presented next.

4.2.1. Camera calibration performance

The first test analyzes the performance between the webcam with its automatic calibration and the industrial camera with the calibration proposed. **Figure 11** presents the ROC curve as well as the area under curve (AUC) for this comparison. The HoG and the Openface algorithms were used with both cameras for detection and recognition, respectively.

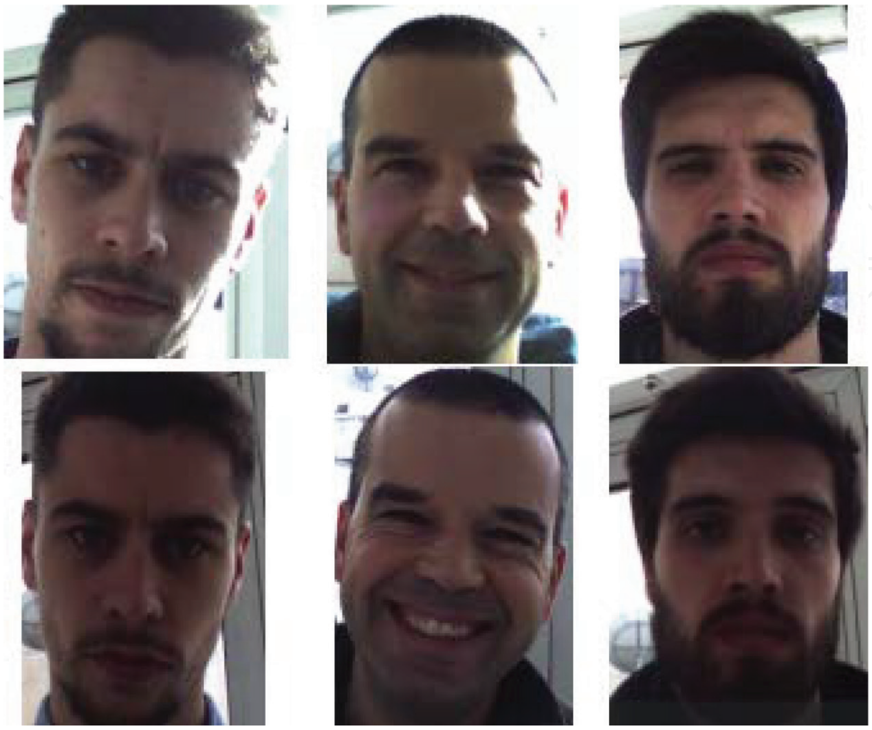

Figure 10. Example of face images acquired in different days with different meteorological conditions.

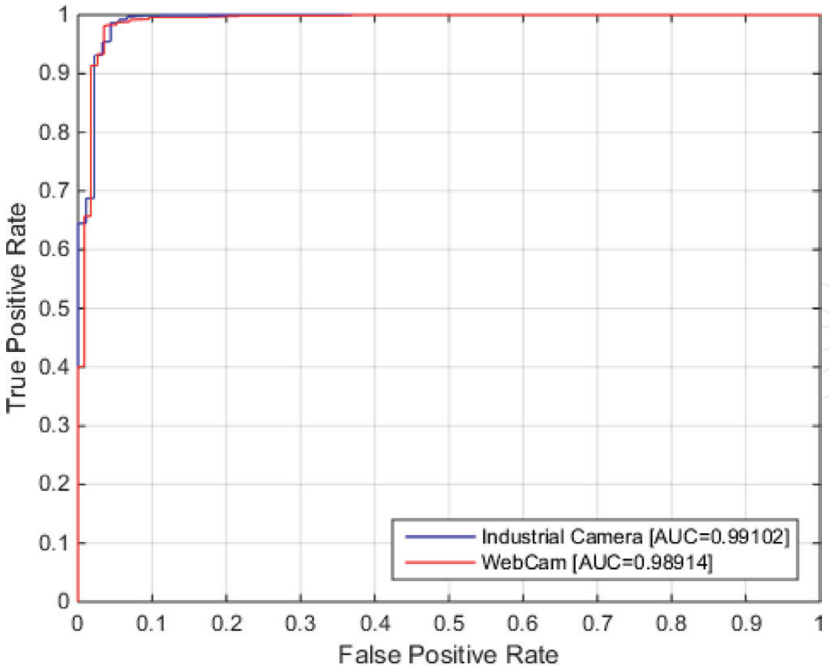

Figure 11. ROC curve comparing the webcam and the industrial camera performance using the same algorithms.

Comparing the AUC resulting from the graphic presented in **Figure 11**, it can be concluded that the proposed method using the industrial camera is better than webcam.

4.2.2. Face detection performance

In this section, the performance of the HoG and MTCNN face detection algorithms is presented. The time that it takes to detect faces in images with dimensions of 752×480 pixels was first measured. Posteriorly, the accuracy of each algorithm using a video recorded at the time of the tests was tested. **Table 1** provides the results for both algorithms.

Analyzing this table, although the HoG algorithm has a lower processing time and less false positives, the MTCNN algorithm has the advantage of detecting faces in profile view and consequently detect more faces for the input of the face verification stage.

4.2.3. Face recognition and preprocessing algorithms performance

Results of the performance of the recognition algorithms tested with and without the preprocessing methods of gamma correction and CLAHE are presented here. As all algorithms are based on neural networks, it is important to point out that, despite using a specific

	HoG	**MTCNN**
Processing time (ms)	60	121
Total detections	592	740
False positives	1	8

Table 1. Processing times, total detections, and false positives for each face detection algorithm.

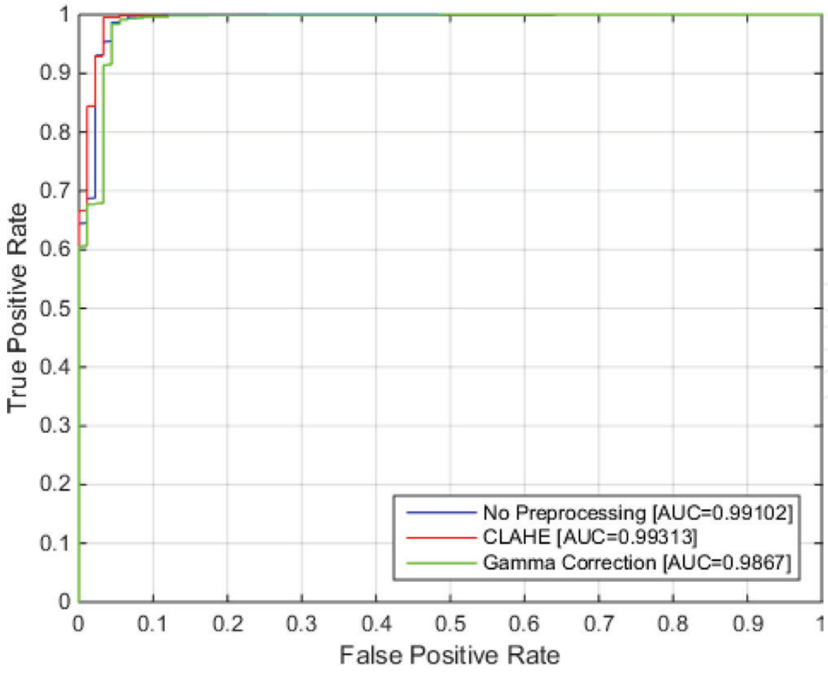

Figure 12. ROC curve presenting the performance of OpenFace using CLAHE, gamma, and no preprocessing methods.

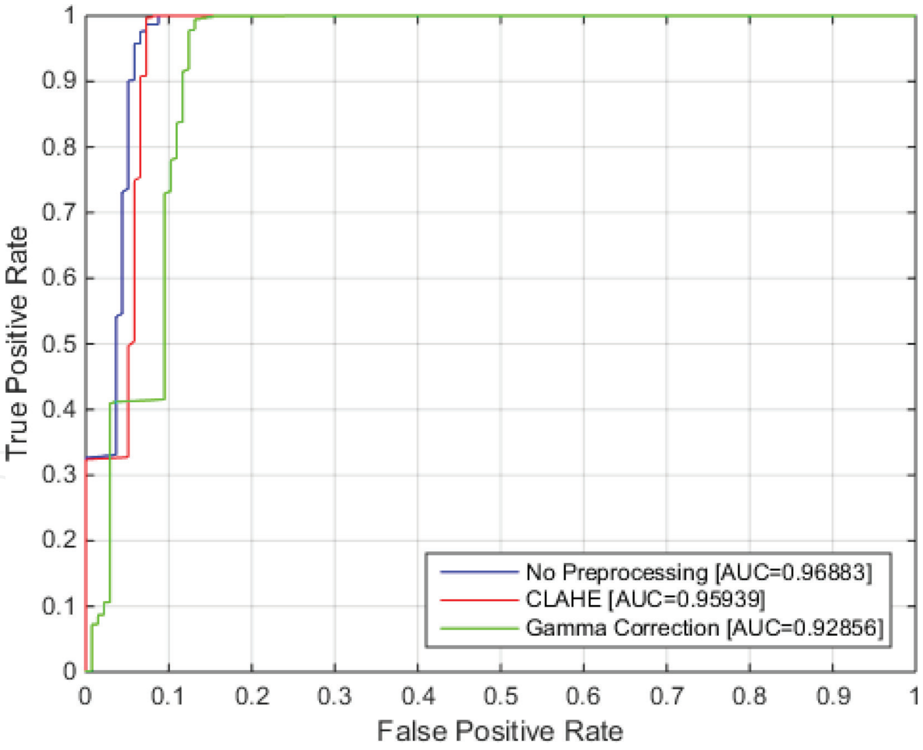

Figure 13. ROC curve presenting the performance of DML using CLAHE, gamma, and no preprocessing methods.

preprocessing method, the network was not retrained. The results might improve if the preprocessing methods are applied to the images that are used to train the neural network.

Figures 12–14 present the algorithms performance using no preprocessing algorithms and comparing its results with the use of CLAHE and gamma correction.

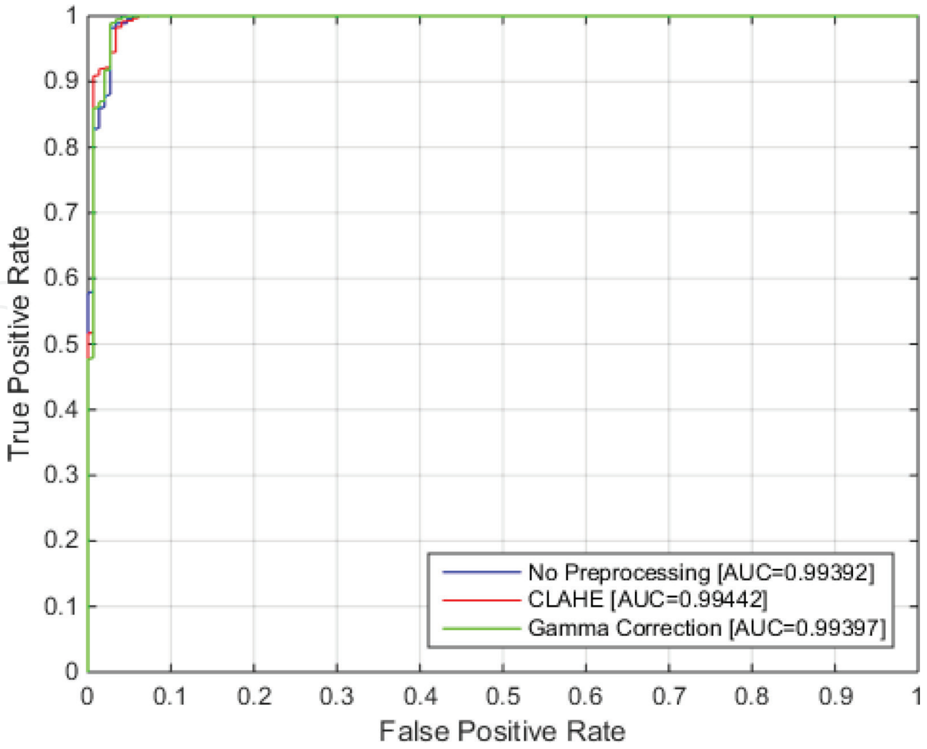

Figure 14. ROC curve presenting the performance of DeepFace using CLAHE, gamma, and no preprocessing methods.

	OpenFace	DML	DeepFace
Forward network runtime (ms)	236	293	110

Table 2. Processing times for forward pass in each network.

Table 2 shows the processing time that takes each face image to forward pass the neural network of each algorithm.

The results obtained using the proposed algorithms for face recognition show that the gamma correction revealed to have a negative impact in OpenFace and DML algorithms. As for DeepFace algorithm, the gamma correction does not have a significant impact on the output.

The preprocessing algorithm CLAHE has a positive impact in all face recognition algorithms used in this work.

Through the analysis in **Figures 12–14** and **Table 2**, the DeepFace is the better algorithm for using in the face verification stage. The DeepFace is two times faster than the other proposed algorithms and the AUC is higher.

5. Conclusion

This chapter presented a face verification solution and studies where algorithms and cameras are appropriately used under uncontrolled environments. Regarding the camera and its calibration, the industrial camera had a better performance compared to the webcam as the calibration method presented focus on the best face image that can be acquired. As for software, both detection algorithms presented a good performance. Despite that, MTCNN seems to have the best performance as it detects faces where subject is in the profile view. In relation to the recognition and the preprocessing algorithms, CLAHE algorithm had a positive impact in all the recognition algorithms as for the gamma correction had a negative impact. It is believed that the results would improve if the preprocessing technique was applied in all the face images used for the training of the neural network. Unfortunately, the training of these types of neural networks takes over a day using powerful GPUs which are difficult to access. Despite that, the overall performance of the system was satisfactory and, from now on and according to the experiments, the best solution for this system is the use of an industrial camera, MTCNN for face detection, CLAHE for preprocessing, and DeepFace for the face verification stage.

The future work goes through the implementation of the solution in larger scales where more people would use it. Until then, the training of new neural networks using the preprocessing techniques is presented, and the study of new alternatives for cameras is on the agenda.

Acknowledgements

This work is partially funded by National Funds through the FCT—Foundation for Science and Technology in the context of the project UID/CEC/00127/2013.

Author details

Ricardo Ribeiro*, Daniel Lopes and António Neves

*Address all correspondence to: rfribeiro@ua.pt

Institute of Electronics and Informatics Engineering of Aveiro, Aveiro, Portugal

References

[1] Zhao W, Chellappa R, Phillips PJ, Rosenfeld A. Face recognition: A literature survey. ACM Computing Surveys. 2003;**35**(4):399-458

[2] Amos B, Ludwiczuk B, Satyanarayanan M. OpenFace: A general-purpose face recognition library with mobile applications. Technical Report. CMU-CS-16-118. CMU School of Computer Science; 2016

[3] Jafri R, Arabnia HR. A survey of face recognition techniques. Journal of Information Processing Systems. 2009;**5**(2):41-68

[4] Olszewska JI. Automated face recognition: Challenges and solutions. In: Ramakrishnan S, editor. Pattern Recognition - Analysis and Applications, Chapter 4. Rijeka: InTech; 2016

[5] Neves AJR, Cunha B, Pinho AJ, Pinheiro I. Autonomous configuration of parameters in robotic digital cameras. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2009. pp. 80-87

[6] Wood R, Olszewska JI. Lighting-variable AdaBoost based on system for robust face detection. In: Proceedings of the 5th International Conference on Bio-Inspired Systems and Signal Processing; SciTePress Digital Library; Algarve, Portugal. 2012. pp. 494-497

[7] Lopes D, Neves A. A study on face identification for an outdoor identity verification system. Vol. 27. Lecture Notes in Computational Vision and Biomechanics. 2017. pp. 689-699

[8] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: Proceedings–2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005; 2005. pp. 886-893

[9] Zhang K, Zhang Z, Li Z, Qiao Y. Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters. 2016;**23**(10):1499-1503

[10] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) [Internet]. 2016. pp. 770-778. Available from: http://ieeexplore.ieee.org/document/7780459/

[11] Huang GB, Ramesh M, Berg T, Learned-Miller E. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. University of Massachusetts Amherst Technical Report. 2007;**1**:07-49

[12] Ng HW, Winkler S. A data-driven approach to cleaning large face datasets. In: 2014 IEEE International Conference on Image Processing, ICIP 2014; 2014. pp. 343-347

[13] Parkhi OM, Vedaldi A, Zisserman A. Deep face recognition. In: Procedings of the British Machine Vision Conference 2015 [Internet]. 2015. pp. 41.1-41.12. Available from: http://www.bmva.org/bmvc/2015/papers/paper041/index.html

[14] Yi D, Lei Z, Liao S, Li SZ. Learning Face Representation from Scratch. 2014. Available from: http://arxiv.org/abs/1411.7923

[15] Schroff F, Kalenichenko D, Philbin J. FaceNet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2015. pp. 815-823

[16] Wen Y, Zhang K, Li Z, Qiao Y. A discriminative feature learning approach for deep face recognition. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2016. pp. 499-515

[17] Engineering C. Gamma correction technique based feature extraction for face recognition system. International Journal of Computational Intelligence and Informatics. 2013;**3**(1):20-26

[18] Bebis G, Boyle R, Parvin B, Koracin D, Remagnino P, Nefian A, et al. Illumination normalization for color face images. Advances in Visual Computing. 2006;**4291**(2):90-101. Available from: http://www.springerlink.com/content/y74g332844240075/

[19] Zuiderveld K. Contrast Limited Adaptive Histogram Equalization. In: Graphics Gems. Academic Press; 1994;474-485. ISBN: 9780123361561. Avaliable from: http://www.sciencedirect.com/science/article/pii/B9780123361561500616

[20] Pizer SM, Amburn EP, Austin JD, Cromartie R, Geselowitz A, Greer T, et al. Adaptive histogram equalization and its variations. Comput Vision, Graph Image Process. 1987; **39**(3):355-368

[21] Benitez-Garcia G, Olivares-Mercado J, Aguilar-Torres G, Sanchez-Perez G, Perez-Meana H. Face identification based on contrast limited adaptive histogram equalization (CLAHE). In: Proceedings of International Conference on Image Processing, Computer Vision and Pattern Recognition; 2011

[22] Arcoverde Neto EN, Barreto RM, Duarte RM, Magalhaes JP, Bastos CCM, Ren TI, et al. Real-time head pose estimation for mobile devices. Integrated Computer-Aided Engineering. 2014;**7435**:467-474. DOI: 10.1007/978-3-642-32639-4_57%5Cn