Computer Vision
False face detection in the Tracking

Henry Cruz, Juan Meneses
Centro de Investigación en Tecnologías de Software y Sistemas Multimedia para la Sostenibilidad (CITSEM), Universidad Politécnica de Madrid
Madrid, España
henry.cruz@upm.es
juan.meneses@upm.es

Martina Eckert, José F. Martinez
Centro de Investigación en Tecnologías de Software y Sistemas Multimedia para la Sostenibilidad (CITSEM), Universidad Politécnica de Madrid
Madrid, España
martina.eckert@upm.es
jf.martinez@upm.es

***Abstract.* — The following work focuses on the tracking process for obtain face detection and tracking of faces, as part of the computer vision. To complete this study, the tracking and facial recognition process by the software tool MATLAB and the Viola-Jones algorithm will be developed. Once the codification has been developed, the results are assessed and the performance of the created script has been verified. Also an important factor for mention is the control of rate error of the false faces recognition. Also we show a methodology for evaluate the algorithms efficiency. Finally the different conclusions are made in relation the results obtained.**

Keywords; Real time, face detection, tracking, face recognition, Viola-Jones, Gaussian filter.

I. INTRODUCTION

The computer vision has different components such as; detection, identification, tracking, localization, processing, display, all this components interact with each other. The tracking into this process is a necessary component.

For other side the tracking of faces is an application gives us real information for multiple applications [1, 2, 3]. In order to do the tracking of faces, the development of different phases in the treatment of the images is needed also an organized sequence is set before the tracking is developed [4, 5]. The first one is the identification that is the extraction of information of the environment through a camera, then the face is detected in the image and finally, the face is tracked by the camera, what means the camera follows the face [6, 7].

The areas of investigation in which the facial identification and recognition are varied: biometry, the safety of information, the accomplishment and vigilance of law, interaction human being- machine, access control, photo cameras, computers, patterns recognition, neuronal nets and psychology in which facial expressions, gesture perceptions and emotions interpretation, augmented reality, etc.[8].

II. VIOLA JONES FOR FACIAL RECOGNITION

Detection, identification and tracking process has a similar structure [9]; it shows in the figure 1. After preprocessing, is used the Viola-Jones filter. In order to detect the faces and their facial components in the image taken, an image processing must be done, it is used in face detection in colour images [10].

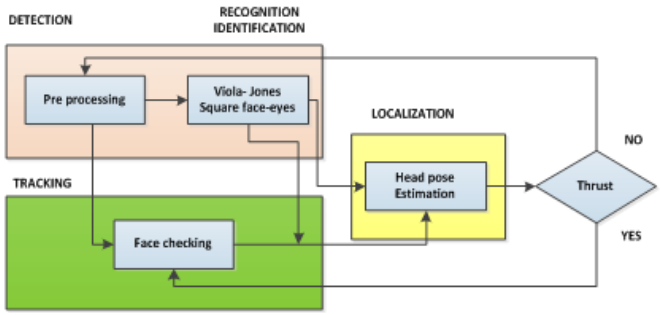

Figure 1. Detection, recognition, tracking and localization process of faces.

The Viola-Jones algorithm can be used, for normalizes the face both geometric and photometric to balance particular unwanted variations in the lighting conditions (intensity, colour, temperature, direction, etc.) and face geometry (scale, pose, orientation, etc.) [11, 12]. This algorithm can find the important facial characteristics (eyes, nose and mouth) in each fixed area through the algorithm.

In order to apply the algorithm described, the rectangles shown in the Fig. 2 are used among a variety of parallelogram rectangles. Each characteristic corresponded to the difference of the sum of the pixels in the white and black rectangles [13].

Figure 2. Rectangles used in the Viola-Jones algorithm.

III. DETECTION PROCESS IN FACIAL RECOGNITION

A. *Use of libraries for detection.*

In order to do face detection, is necessary use the Viola-Jones algorithm through the library *vision*, more specifically *vision Cascade Object Detector*, the unique library based on objects recognition. Then a picture is taken and compared with the before one and the following one, it is defined as a cascade of sorters which is explored by whole image in multiple scales and locations. Through the chosen algorithm, the function, named step, obtains the answer for the LTI (Linear Time Invariant). With this, the area of the square where the rectangle is centred is obtained and showed on the image taken.

If two or more faces are wanted to be obtained, again the function *step* is called, using as parameters the image taken, the detected faces and the obtained rectangles through the function *vision shape inserter* to square them again and show them. In this phase the different faces on an image have been detected. This action must be done in continuous time to have an interaction between the program and the user in real time. The Fig. 3 shows the result of the interaction.

Figure 3. Detection result

B. *Use the Gaussian filter for image substracted.*

In order to detect a face, a comparison among several images is done. Firstly, the intensity of the image to be processed is obtained, it is named raw. Then, the media among the images taken is calculated and the common background of the images sequence, named background subtracted, is subtracted. After that, a Gaussian filter, known as Gaussian smoothed [14], is applied and an image without background and noise is obtained. Finally, the concrete threshold is obtained, what is the face desired as shown in Fig. 4.

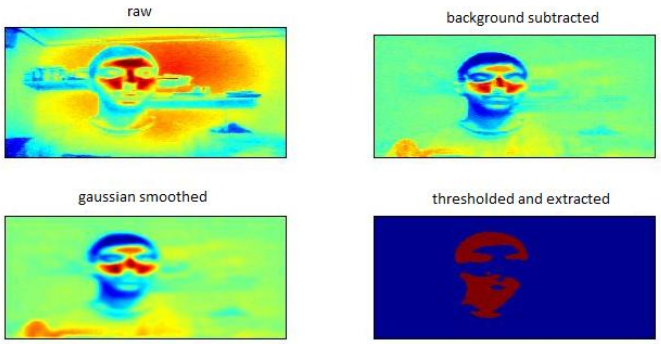

Figure 4. Threshold and extracted process [15]

IV. FALSE FACES INTO TRACKING PROCESS .

At the tracking process, to implement a loop to take the different sequences capture to be processed [16]. Therefore, a number of pictures are chosen to be taken, if do not, the program would be executed up to the infinite, until the buffer would be full or up to the own user would decide to end the execution of the programing environment.

Is necessary to take into consideration that the camera is turned on whilst the script is being executed because otherwise, the programing environment would have to restart. For this reason, a *try-catch* control exceptions has been incorporated to the script.

The script developed allows obtaining the number of false faces (detections treated as faces when in fact they are not) and the total of faces along with their percentages from the number of faces in the script and the number of frames in the video sequence introduced by the user.

To detect a false face, the number of faces obtained has to be higher than the faces that can be processed in the image or a face is detected where there is no face. In (1) efficiency is the percentage of faces rightly detected and the false faces are the faces wrongly detected and are mistakes in (2).

$$Efficiency\ (\%) = \frac{Number\ of\ faces\ detected\ rightly}{Frames} \cdot 100 \qquad (1)$$

$$Mistakes\ (\%) = \frac{Fakes}{Faces\ to\ be\ detected \cdot Frames} \cdot 100 \qquad (2)$$

If the efficiency and mistakes are summed up, the result is not 100% because when a mistake is evaluated, the faces in the shot are taken into account and more one false face can be shown in the same image. In the equations (3) and (4) shows the formulas to be used if there are no faces to be detected on the image.

$$Efficiency\ (\%) = \frac{Frames - Mistakes}{Frames} \cdot 100 \quad (3)$$

$$Mistakes\ (\%) = \frac{False\ Faces}{Frames} \cdot 100 \quad (4)$$

V. EXPERIMENTATION AND RESULTS FOR FALSE FACES ALGORITHM.

In the experimentation, we have made twelve tests, considering different aspects that generate errors or false faces. These aspects are in relation to; pose estimation, occlusions condition, camera distance, light intensity, colour intensity. The take this data permit us, evaluate the algorithm quality through proposal evaluation methodology. The results allow us know the efficiency rate and errors in each frame into faces detection in real time and for each test.

A. Test for the evaluation.

The tests realized to assess the efficiency of the script are the following:

- Test 1: 1 face from 1 meter.
- Test 2: 1 face from 2 meters.
- Test 3: 1 face from 5 meters.
- Test 4:2 faces from 1 meter.
- Test 5:2 faces from 2 meters.
- Test 6:2 faces from 5 meters.
- Test 7:1 face from 1 meter and 1 face from 2 meters.
- Test 8:1 face from 1 meter and 1 face from 5 meters.
- Test 9:1 face from 2 meters and 1 face from 5 meters.
- Test 10:1 face (an eye covered) from 1 meter.
- Test 11:1 face (an eye covered) from 2 meters.
- Test12:1 face from less than 1 meter (an eye covered).

The frame rate or number of photographs taken by second should be 50 Hz minimum to the human eye detects continuity in the video and does not detect a video sequence of a lot of photographs. The frame rate can be obtained through the function *getdata*, that returns the number of frames and the time passed on the video, and calculates it through the media of the time difference between each image capture. In this case the frame rate obtained is around 14 Hz. Once this process has been done, the option to extract the video sequence has been implemented in a file. The extraction has been done through the functions move in *getframe* and *move2avi*.

B. Test detection of faces result.

Thanks extract the video, the faces found have been detected during different instants of determinate time and the tracking is completed. In the following box, the results in percentages of the number of recognized faces and false faces are showed along with the faces recognized rightly and the false faces. For the tests there have been taken 100 shots of images with a lapse of time or frame rate of 15-20 seconds. As a result, the comparative consideration can be summarized as presented in TABLE I.

TABLE I. TEST DETECTION OF FACES RESULT

Frames = 100	Faces obtained	Right faces (%)	False faces	False faces (%)
Test 1	101	99,00	1	1,00
Test 2	101	99,00	1	1,00
Test 3	101	95,00	3	3,00
Test 4	200	100	0	0,00
Test 5	205	95,00	5	2,50
Test 6	202	99,00	2	1,00
Test 7	200	100	0	0,00
Test 8	200	100	0	0,00
Test 9	210	90,00	10	5,00
Test 10	75	71,00	2	2,00
Test 11	102	98,00	2	2,00
Test 12	82	78,00	2	2,00

One of the occurred phenomenon is the detection of false faces. In these cases, the program detects a face where there is no one as shown in Fig. 5.

Figure 5. Detection of a false face

Undertaking the efficiency tests of the script, it is observed that the majority of the false faces come from the mix of red and white colours as shown in Fig. 6.

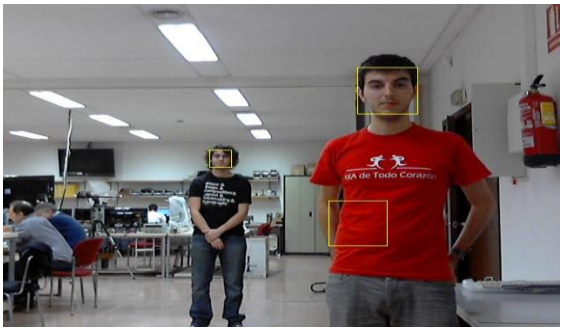

Figure 6. Detection of a false face on reds

It is interesting to highlight that when an only face is treated in normal conditions (looking at the front of the camera with simple movement to the sides), the efficiency of the program is very high. Because of that, tests 1 and 2 show these high percentages. However, if test 3 is considered, the efficiency is affected but still is quite high. Fig. 7 shows it.

Figure 7. Detection false face from an appreciable distance.

In the recognition of two faces, it can be observed that the script has good results, too. Test 4, 5 and 6 are an example of that. In this case, because the distance is longer, some more false faces appear, because of the reds, but there are no relevant as showed in Fig. 7.

In the following tests (7, 8 and 9), there is a variation among the faces to be recognized. In this case, the efficiency is high, although in the last test more false faces appear some of them because of the reds explained above and others for the dependency among the subjects who do the test as shown in Fig. 8.

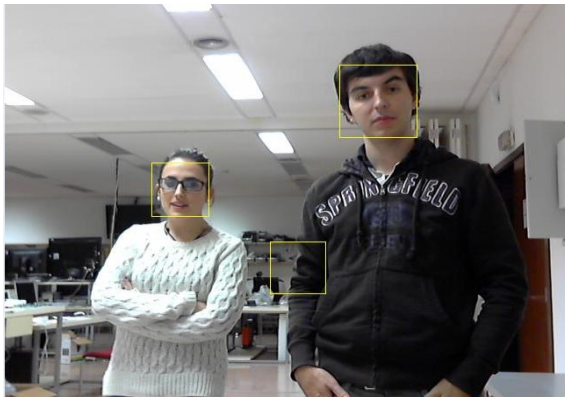

Figure 8. Detection of a false face by correlation between subjects

The script is analysed for a subject covering one eye as shown in test 10. There are some difficulties to detect the face in the beginning but after some time and thanks the estimations calculates by the algorithm, the face is tracked accurately. However, keeping a longer distance from the camera, as shown in test 11, the detection is almost immediate and perfect and the efficiency is high as shown in Fig. 9.

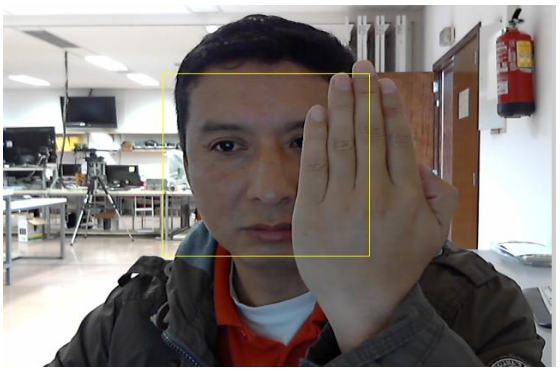

Figure 9. Detection of a face partially covered.

In the last test, the script is assessed very close to the camera. With that, it can be said that if the forehead or the chip is partially covered, the subject is not recognised fully as shown in Fig. 10.

Figure 10. Errors face detection by proximity.

The efficiency rate of the faces obtained vs. false faces is higher how show in the Fig. 11. In this way the faces detected are almost 98 per cent in relation with almost 2 per cent of false faces for all tests. It demonstrates in general the efficiency of this algorithm.

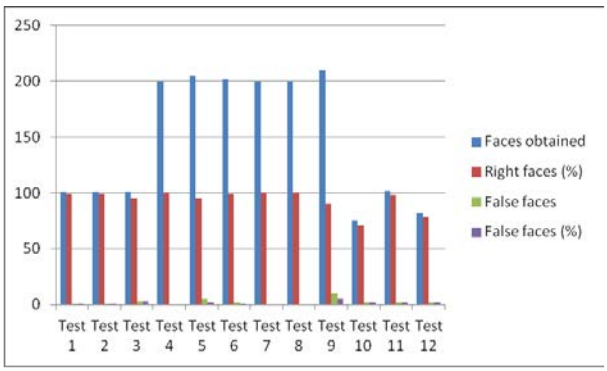

Figure 11. Comparison between faces obtained and false faces for each test.

VI. CONCLUSIONS

The best position to recognise the faces on the image is to be located perpendicularly to it. Through an algorithm the face will be detected properly although movements occur. There are occasions in which false faces are detected on the taken images. These detections are no faces but they have been processed as ones. The majority of the false faces are produced by the mix of reds and whites.

The efficiency of the script is higher if the faces are located a moderate distance. In some occasions false faces are detected by the correlation of subjects on the same image. When the program detects a face, it takes certain range searching around the face in order to provide better results. To detect a face, the program takes into account mainly the forehead and the chin of the subject. In general we have almost 98%

The evaluation methodology used, allow us to know the real efficiency rates of this algorithm. This methodology can be used too, for evaluate other tracking methods as well as targets detection.

VII. FUTURE WORK.

As a consequence of the continuous development of the smartphones and the increase in the speed and reduction on the size of the microprocessors, the implementation of software in Open CV is being studied.

Thanks to the great amount of libraries already created that can recognise the patters existing in the environment such as a pavement, a road and can help drivers to improve their efficiency behind the wheel. This software can be implemented through MATLAB/OCTAVE, C++ or Java adding functions as the accelerometer in the smartphone or the system of location incorporated. In case of a programing difficulty in the smartphone, a micro controller with a camera can be used. This algorithm also can be used for detection in crowded scenarios as well as multiple objectives detection.

ACKNOWLEDGMENT

This work was sponsored by Center on Software Technologies and Multimedia Systems for Sustainability (CITSEM) of the Technical University of Madrid.

Henry Cruz Carrillo gives thanks the Government of Ecuador for the scholarship awarded through “Secretaría Nacional de Educación Superior Ciencia Tecnología e Innovación” (SENESCYT).

REFERENCES

[1] B. Woodrow, T. Caudell, “Fundamentos de Informática usable y Realidad Aumentada”, Edit. Mahwah, New Jersey, 2009, pp 107-123.

[2] J. Lee , D. Seo and Rhee G, “Visualization and interaction of pervasive services using context-aware augmented reality”, Expert Systems with Applications Journal,Vol. 35, Issue 4, pp. 1873‑1882 , 2008.

[3] P. Milgram , et al., “Augmented Reality: A class of displays on the reality-virtuality continuum”, Conference on Telemanipulator and Telepresence Technologies, Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE), Vol. 2351, pp. 282-292, Boston, 1994.

[4] C. Gámez, V. Villena, “Diseño y desarrollo de un sistema de reconocimiento de caras”, Tesis de grado, Universidad Carlos III de Madrid, 2009.

[5] T. Herrero, J. Villena, “Sistema automático de detección y etiquetado de caras en imágenes”, Tesis doctoral, Universidad Carlos III de Madrid ,Madrid, 2010.

[6] Al-Mohair et al., “Human Skin Color Detection: A Review On Neural Network Perspective”, International Journal of Innovative Computing Information and Control, Vol. 8, Issue 12, pp. 8115-8131, 2012.

[7] D. Ngan, et al., “Face segmentation using skin color map in videophone applications”, IEEE Transactions on Circuits and Systems for Video Technology Journal, Vol. 9, Issue 4, pp. 551-564, Jun 1999.

[8] A. Cheddad, et al., “A skin tone detection algorithm for an adaptive approach to steganography”, Signal Processing Journal. Vol. 89, Issue 12, pp. 2465-2478, Dec 2009.

[9] M. Perreira, et al., “Fast, low resource, head detection and tracking for interactive applications”, PsychNology Journal, Vol 7, pp. 243-264 , 2009.

[10] R. Hsu ,M. Mottaleb and A. Jain, “Rule-Based Face Detection in Color Images using Normalized RGB Color Space - A Comparative Study”. Proceddings 3rd IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), pp.446-450 , 2012.

[11] P. Viola, M. Jones “Rapid object detection using a boosted cascade of simple features”, Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, pp. 511-518, 2001.

[12] P. Viola, M. Jones, “Robust Real-time object detection”, International Journal of Computer Vision, Issue 57, Vol. 2, pp. 137-154, 2004.

[13] I. Landesa , J. Alba, “Detección de caras y localización de características faciales para reconocimiento biométrico, Tesis Doctoral, Universidad de de Vigo, Vigo-España, 2003.

[14] J. B. Hayet , Introducción al uso del filtrado de Kalman , Edit. CIMAT, Centro de Investigacion en Matemáticas, CONACYT, Mexico, 2008, pp. 3-34.

[15] S. Dave, “Tutorial Kalman filter with MATLAB code”, Online video Guidance, Dec 2012.

[16] D. Guequan et al. “Scene Aware Detection and Block Assignment Tracking in crowded scenes”. Image and Vision Computing Journal, Vol 30, Issue 4-5, pp. 292-305, May 2012.