UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Visión Electrónica
Más que un estado sólido
https://doi.org/10.14483/issn.2248-4728

VISIÓN ELECTRÓNICA

A CASE-STUDY VISION

Telematic artificial vision prototype for recognition of characteristics in people and vehicles

Prototipo telemático de visión artificial para el reconocimiento de características en personas y vehículos

Fanny Suárez-Mosquera [1], *Cristian Dangelis Ballén-Martínez* [2], *Gerardo Alberto Castang-Montiel* [3]

INFORMACIÓN DEL ARTÍCULO

Historia del artículo:
Enviado: 22/01/2021
Recibido: 28/01/2021
Aceptado: 17/03/2021

Keywords:
Arduino
Artificial
Characteristics
Facial
License plate
Ocr
Prototype
Recognition.

Palabras clave:
Arduino
Artificial
Características
Facial
Placas
Ocr
Prototipo
Reconocimiento.

ABSTRACT

In a world increasingly connected with technology and with the growing need for security in all aspects of life, this project arises where the problem of security in educational establishments will be addressed, but it can be used in many areas, such as in hospitals, parking lots, securities transporters, airports, etc. This prototype aims to recognize patterns (Vehicle Plates) and characteristics (Facial Patterns), one of the main premises of the current project was the use of free software, to create a successful and easy-to-use solution.

RESUMEN

En un mundo cada vez más conectado con la tecnología y con la creciente necesidad de seguridad en todos los aspectos de la vida, surge este proyecto donde se abordará el problema de seguridad en los establecimientos educativos, pero que puede ser utilizado en muchos ámbitos, como en hospitales, parqueaderos, transportadoras de valores, aeropuertos, entidades militares, etc. Este prototipo tiene como objetivo el reconocimiento de patrones (Placas Vehiculares) y de características (Patrones Faciales), una de las premisas principales del actual proyecto fue la utilización de software libre, para crear una solución acertada y de fácil utilización.

[1] BSc. in Telematic Engineering, Universidad Distrital Francisco José de Caldas, Colombia. E-mail: fsuarezm@correo.udistrital.edu.co

[2] BSc. in Telematic Engineering, Universidad Distrital Francisco José de Caldas, Colombia. E-mail: cdballenm@correo.udistrital.edu.co

[3] BSc. In Electronic Engineering, Universidad Autónoma De Colombia, Colombia. MSc. in Teleinformatics, Universidad Distrital Francisco José de Caldas, Colombia. E-mail: gacastangm@udistrital.edu.co

Cite this article as: F. Suárez-Mosquera, C. D. Ballén-Martínez, G. A. Castang-Montiel, "Telematic artificial vision prototype for recognition of characteristics in people and vehicles", *Visión Electrónica*, vol. 15, no. 1, pp. 104-112, 2021. https://doi.org/10.14483/22484728.17981

1. Introduction

The prototype described below was created to provide an accurate and easy solution. With this prototype different schools can learn about miscellaneous drivers and vehicles that provide school route service, providing a validation tool that can be integrated with Secretariat of Mobility registration system. A recognition of facial patterns and features can be made, all of this supported by free software as Open Alpr and Face recognition libraries, the last one through 128-point feature vectors, which makes face representation for later comparison. Allowing it to be easy to use and a low-cost solution, also it can be used in a number of scenarios making it highly scalable. Enhancing the centralization of data by using a distributed database, generating high availability and a higher security level.

2. Analysis stage

As part of this stage, an analysis of the chosen hardware and software architecture is carried out, and the scope and limitations of this are defined for subsequent implementation.

2.1. Objective

Design an artificial vision prototype for facial recognition of drivers and school route plates for entry and/or exit from the institution.

2.2. Theorical frame

For a better understanding and knowledge of the project, the most relevant terms will be highlighted:

2.2.1. Open ALPR

OpenALPR is an open source library used in automatic license plate pattern recognition written in C++ with links in C#, Java, Node.js, Go and Python. The library analyzes images and video sequences to identify license plate patterns.

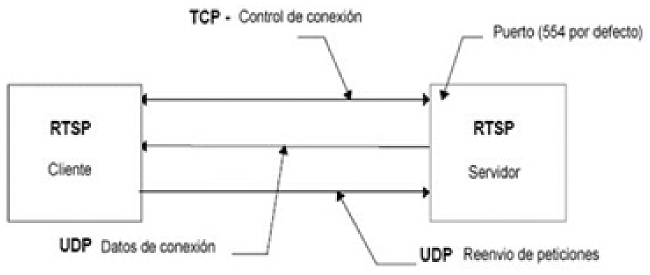
2.2.2. OPENCV

OpenCV is a free machine vision library originally developed by Intel. It contains more than 500 functions that cover a wide range of areas in the vision process, such as object recognition (facial recognition), camera calibration, stereo vision and robotic vision

2.2.3. RTSP (Real Time Streaming Protocol)

RTSP is a non-connection-oriented protocol, instead the server maintains a session associated with an identifier, in most cases RTSP uses TCP for player control data and UDP for audio and video data.

Figure. 1. RTSP Communication. [1]



3. Design stage

In this stage of the methodology, the recognition of the involved agents with the system is developed and in addition how is the interaction between these with the prototype, besides relating the model of hardware and software of the system, for this it is divided in the Software component and Hardware component.

3.1. Hardware model

In this project, this item specifically is highly scalable and configurable, since inside this prototype it was found the need to build a hardware component, which would allow the driver to apply to the institution, which will start the validation circuit of the telematic prototype (IP camera and the rest of the services of facial recognition and registration).

This component is made up of a circuit with an Arduino card, a protoboard and a button, but given the capacity and the interaction that it can have with the system, it could be easily adapted to systems of sensors, fences, automatic doors, registers and other access devices.

3.2. Software model

As it was indicated before this prototype handled two fronts between them; the software that will be the interaction of the system with the final user, this component is the most important of the project, the system has components of graphic interface where the administration modules and the configuration modules will be visualized, besides seeing the alerts with each one of the different entries that occur.

This was developed with free and open-source technologies such as Python, Java, JSF, Java programming languages were used, besides using free distribution tools for its development such as NetBeans, notepad ++, Visual Studio Code, and Mysql database engine, Maria DB.

3.2.1. Database model

The database modeling is perhaps the most important point in the software development, since it is from this design that the system's planning and interaction begins, in addition to managing the distributed database, to maintain the reliability and availability standards required for proper operation. For this process, the MARIA DB database engine was used, with a GPL license, in which the configuration was carried out in order to establish the distributed database.

3.2.2. System agents

User interactions were defined for this system, but as indicated above the specifications may change according to the environment to be used, either with the Hardware/Software separation or the set of both. The following agents were defined in this section.

Table 1. Definition of system agents.

Agent	Description
Admin	Responsible for the administration of alerts, personnel and vehicles authorized to enter the institution.
Watchman	Agent who receives notifications of unauthorised entry into the institution, person responsible for granting or refusing entry into and/or leaving the institution
Driver	Agent that interacts with the system through the Arduino board, for this prototype.

Source: own.

3.2.3. Architecture

In every project it is indispensable to think about a good architecture, because it defines a path to begin to build the system. The way in which the components will interact with other systems and thus achieve the main objective which is to have a functional prototype, ready for implementation.

3.2.3.1. License plate recognition with openALPR

It was used OpenALPR, which is a library that helps the automatic recognition of license plates, this is open source. It operates as follows for the video stream:

1. The image flow will be constantly extracted from the IP camera in MJPEG format through RTSP which is normally done on a video format if the board can be displayed OpenALPR will start the validation.

2. The agent will start the process automatically and ALPR processes the flow as fast as possible while searching for plate images.

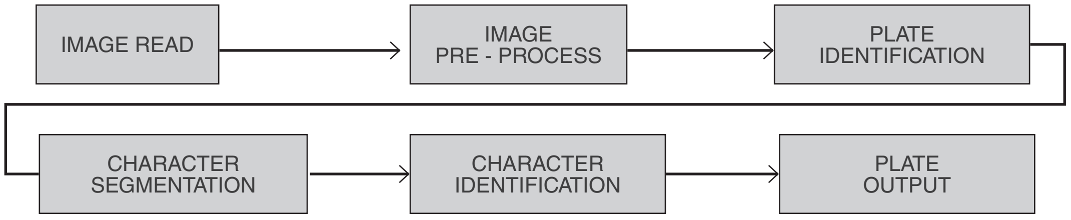
3. When one or more plates are detected, the information will be written to a local beanstalkd queue (pipe name "alprd") as JSON data.

4. Optionally, ALPR will also save the image in a configurable location as a jpeg.

5. ALPR will also run a separate process that will empty the beanstalkd queue and upload the data to the server via HTTP.

Figure 2 below describes the operation of the library.

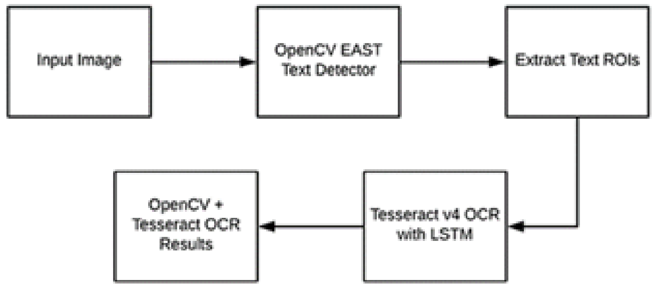
Figure 2. OpenALPR operation.



Source: own.

This library works with Tesseract and OpenCV for character recognition, from version 4, Tesseract to recognize an image containing a single character, uses a convolutional neural network (CNN). For the case of variable length text, they solve it using RNN and LSTM (Long Term Memory) which is learning based recognition.

Figure 3. Pipeline Opencv. [1]


Input Image
OpenCV EAST Text Detector
Extract Text ROIs
Tesseract v4 OCR with LSTM
OpenCV + Tesseract OCR Results

The OpenAlpr operation with an imaging circuit is as follows:

1. To start, is applied OpenCV's EAST (Efficient and Accurate Scene Text Detector) to detect the presence of text in an image. The EAST text detector with help of a neural network will give the delimiter box coordinates (x, y) of the ROI of text.

2. Each of these ROIs is extracted and then passed to the Tesseract v4 LSTM deep learning text recognition algorithm.

3. The output of the LSTM will give the real results.

3.2.3.2. Face recognition

For this implementation was used Face recognition, this library built in python from the Dlib toolkit made in C++ contains deep learning algorithms, which allows to model abstractions providing more accurate results, reaching the model an accuracy up to 98.39%.

Some of the techniques and algorithms used by Dlib are:

- Deep Learning Algorithms (LSTM).

- Optimized general purpose numerical algorithms L-BFGS, Levenberg-Marquardt, BOBYQA among others.

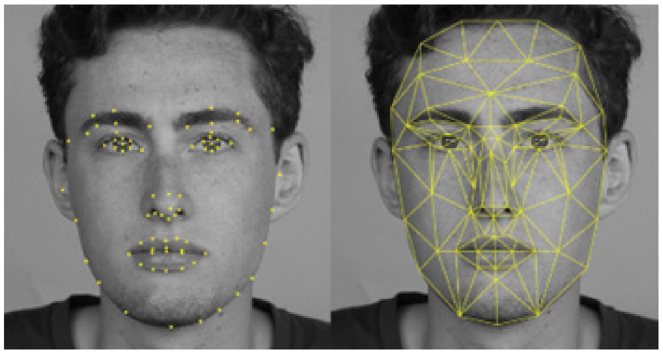
- Graphical model inference algorithms, using Marvok's strings, Bayesian networks within Gibbs sampling.

- Image Processing with the help pf SURF, HOG and FHOG algorithms.

There are different methods for face comparison, such as the waterfall, which analyzes hundreds of small patterns and characteristics that must match, similar to detecting a fingerprint, but this method is not suitable for the present project due to its impossibility of detecting profiled faces or low occlusion.

Others, like the one used in the present project, based on deep metric learning, rely on 128-point characteristic vectors, making a representation of the face, which is similar to what is seen in the following figure.

Figure 4. Dotted vector. [1]



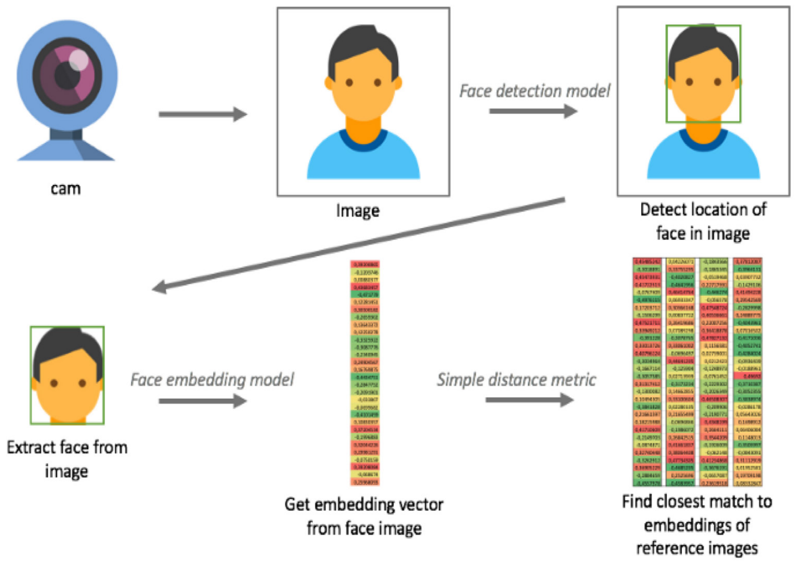
With the given information, the face is identified by the following steps:

1. The face detection model identifies the face's location inside the image.

2. The embedding model is fed to obtain a vector of facial characteristics of size 128.

3. Now it compares this vector with those of its "friends" and finds the most similar.

Figure 5. Recognition cycle. [1]


cam
Image
Face detection model
Detect location of face in image
Extract face from image
Face embedding model
Get embedding vector from face image
Simple distance metric
Find closest match to embeddings of reference images

The algorithms can be up to 99.38% accurate with the right distance threshold between points.

4. Implementation stage

It is at this process stage that the actions were put in place to bring the prototype to a satisfactory conclusion, where the infrastructure for its operation was defined, as shown in Fig. 6 below.

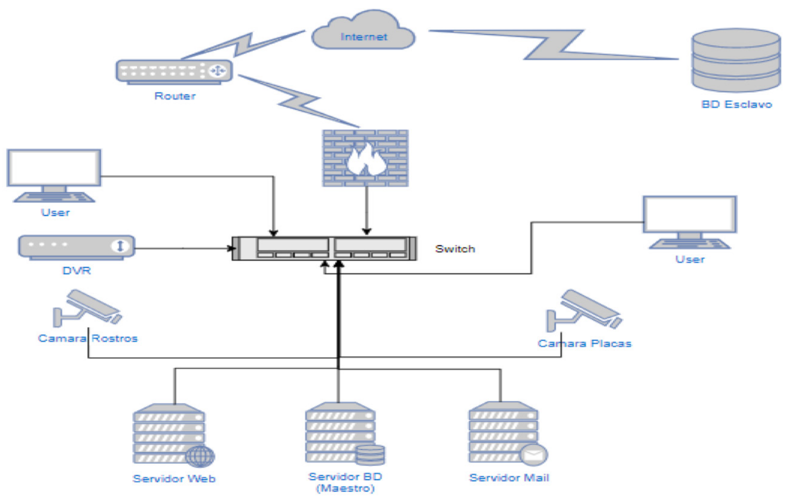
The prototype has a software front that is divided into several components that interoperate with each other such as:

1. Arduino Wiring: This code is stored in Arduino's microcontroller to start the request and begin the whole process. It can be replaced by the access component that the application requires.

2. Python code: It is the backend of the application, where the processing of the images and comparison of these are located.

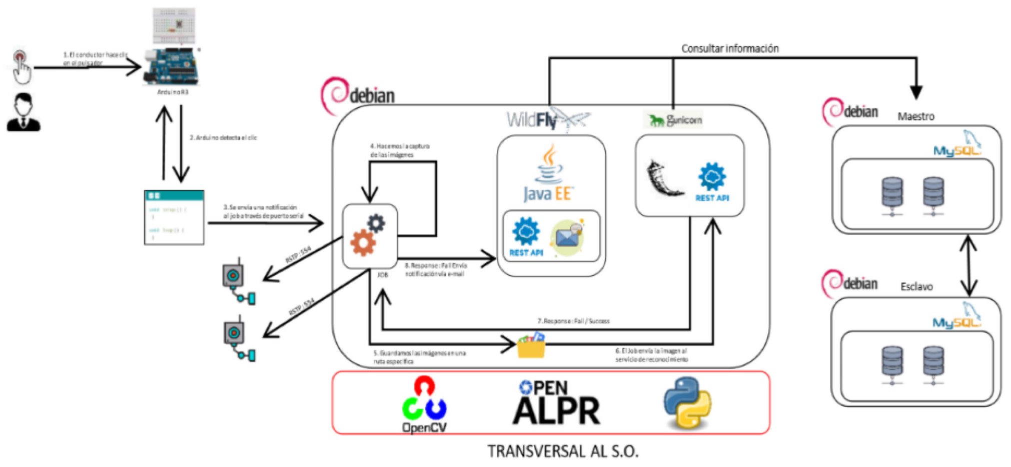
3. Java Code: It is considered the FrontEnd of the telematic system, since it provides a friendly interface to the Administrator and Watchman actors that make up the system.

Figure 6. Implementation.



Source: own.

Figure 7. Prototype overview.


1. El conductor hace clic en el pulsador
Arduino R3
2. Arduino detecta el clic
3. Se envía una notificación al job a través de puerto serial
RSTP :554
RSTP :554
debian
4. Hacemos la captura de las imágenes
JOB
8. Response: Fail Envía notificación vía e-mail
5. Guardamos las imágenes en una ruta específica
7. Response: Fail / Success
6. El Job envía la imagen al servicio de reconocimiento
WildFly
Java EE
REST API
Gunicorn
REST API
Consultar información
debian
Maestro
MySQL
debian
Esclavo
MySQL
OpenCV
OPEN ALPR
TRANSVERSAL AL S.O.

Source: own.

In order to have a better view of the system operation, figure 7 has been validated.

To further understanding, the diagram above describes the technologies used and the flow of information.

1. In the case of the prototype, an Arduino and a push button were used to start the flow, which could be replaced by an access bar or a movement sensor.
2. The Arduino board algorithm processes the message sent through the serial port and sends an alert to the JOB (Automatic Task)
3. It takes a picture of the face and the license plate of the vehicle and stores it in a folder.
4. The JOB (Scheduled Task) sends the 64-based images to the recognition service for analysis.
5. The service generates a response indicating whether the recognition was successful or not to the JOB.
6. If everything was successful the corresponding access is given, otherwise it sends an email to a previously configured account indicating that there was an unauthorized access.

4.1. Hardware/Software integration

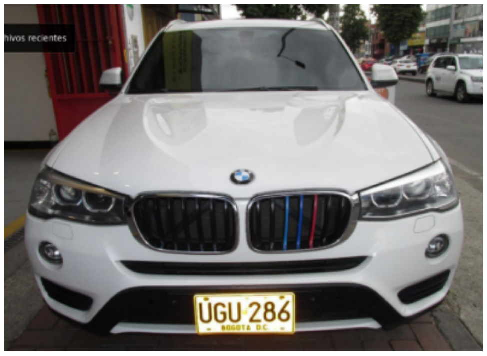
After performing the design, in this stage it is proceeded to make the two components interact with each other and give life to the final product, for this purpose the following elements were defined for its operation.

1. Hardware model (Access component, cameras)
2. Job and Web Services (Communicator between database and application, inserts notifications and validates different faces and plates)
3. Web Application (JEE)

5. Test stage

The accuracy of a project is defined by the test cycle, for this reason, the activities carried out along this stage, aim to identify achievements and limitations obtained in the project, as described below. What is needed for this stage is a vehicle and a person who will carry out the entry simulation to the institution. The first image is the vehicle capture to obtain the plates.

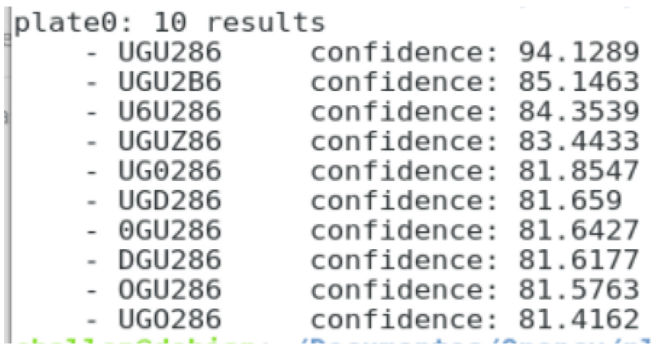
Figure 8. Vehicle plate.



Source: own.

When analyzing the image with the recognition system it is obtained the following results.

Figure 9. Vehicle plate identification.



```
plate0: 10 results
    - UGU286      confidence: 94.1289
    - UGU2B6      confidence: 85.1463
    - U6U286      confidence: 84.3539
    - UGUZ86      confidence: 83.4433
    - UG0286      confidence: 81.8547
    - UGD286      confidence: 81.659
    - 0GU286      confidence: 81.6427
    - DGU286      confidence: 81.6177
    - OGU286      confidence: 81.5763
    - UGO286      confidence: 81.4162
```

Source: own.

A list of 10 possible plates, and a percentage of accuracy, the number of results is parameterizable, it can be obtained from 10 to 50 results. The percentage indicates which characters have more similarity to the image of the vehicle plate so, getting the right plate.

Regarding facial recognition, the first thing to have when starting, are images with people faces that are intended to be identified, ideally getting two profiles of the person and a front photo in order to ensure greater accuracy in recognition, with this, the algorithm is able to recognize whether the face that will be analyzed later is the desired person as shown in Figure 10.

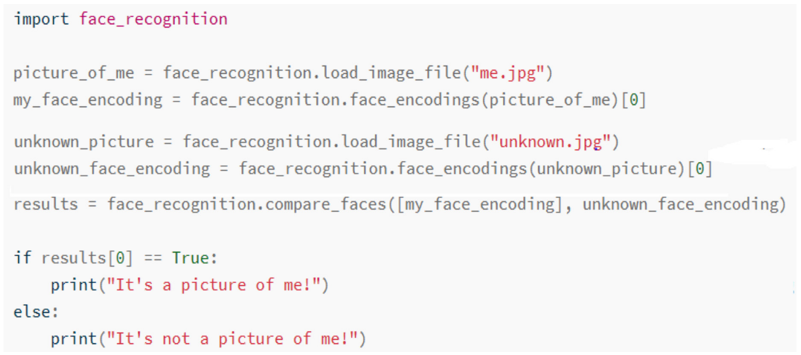
Figure 10. Facial recognition

Source: own.

There are some guidelines that must be taken into account for proper operation, such as the light distance, camera resolution and the number of faces that can appear in the image. The algorithm makes the comparison between previously saved images and the image taken for later identification.

As shown in figure 9, the validation algorithm returns a vector with the analysis response, if true, this means that it recognized a match with one of the base images previously in the case of the faces and validating the sequence of characters for the plates, where each of the steps described in Figure 7 was passed, returning the subsequent validations of having taken the two photos respectively, if the validations are true, It will enter the institution, otherwise the system is notified.

Figure 11. Face Recognition Algorithm.



```python
import face_recognition

picture_of_me = face_recognition.load_image_file("me.jpg")
my_face_encoding = face_recognition.face_encodings(picture_of_me)[0]

unknown_picture = face_recognition.load_image_file("unknown.jpg")
unknown_face_encoding = face_recognition.face_encodings(unknown_picture)[0]

results = face_recognition.compare_faces([my_face_encoding], unknown_face_encoding)

if results[0] == True:
    print("It's a picture of me!")
else:
    print("It's not a picture of me!")
```

Source: own.

6. Closing stage

Over the course of this project, the closing stage was one of the most important, as it was here that the final result was evident, as well as verifying that objectives and test results were met.

The following considerations were made: The prototype's confidence levels may vary, with conditions of distance, comparative (images previously loaded for face validation), luminosity. Recognition levels are high when the prototype has the right photo profiles for optimal comparison.

In the plate case, it was found that the prototype can be highly adaptive since OCR not only recognizes the model of the plates that are handled as standard in Colombia, but also recognizes those parameterized in the system [1].

6.1. *Current works*

To have a more complete overview of the potential of the Open ALPR tool which uses OpenCv, it was researched and it is evident that these libraries have a fairly wide use from implementation of face recognition in major airports, to being a solution for optimal fruit sorting with high quality standards, below are the scenarios.

- **Airports**

In big airports all over the world, the closest and almost imminent option is the face validation of the passengers that enter them daily, like Dulles Washington airport, where a facial recognition system was implemented for the boarding and disembarking of passengers, using learning and deep recognition techniques that are the same of the Opencv library.

This system is highly effective because 3 days after its implementation and operation, the system detected a person carrying a false passport [2], showing the high effectiveness of recognition and the urgency of implementing these systems to strengthen security in airports.

- **Fruit classification**

The applications that have been implemented using this library and variations of others to complement them are open to imagination and the scenarios that are provided, for example, this library has been used for fruit classification as the Tommy mango [3], coffee, where comparisons are made against the ideal to see which ones meet the standards.

6.2. *Future works*

This application is highly scalable and modular to be used in other works, where it can be integrated in a simple way, in this version used in the project or in its paid form the Open Alpr library since this version implements pattern recognition much deeper even detecting plates from various countries, color, vehicle brand and model, even Face recognition can be integrated easily.

Safe Environment and Output Applications:

Being able to create a facial recognition police database integrated with the Distribution solutions of databases, will provide a powerful tool for access to artillery schools, prisons, airports, in addition to the use of OpenCv that detect vehicles and could face a double comparison, detection of fugitives from justice or required by law.

Common Applications:

OpenALPR could be used in malls for plate identification, in security transporters, crew members of the vehicles in charge of transporting money and for validation of the truck in which they are transported, in hospitals for ambulance and personnel identification, in banks and practically everywhere where continuous verification of patterns and characteristics in people and objects is needed.

7. Conclusions

How result of this investigation and the implementation of prototype, is possible concluded that software and libraries allow create solutions for facilitate some challenges modern world.

In the present project the results in the test stage were excellent, having present that for face recognition only a pair failed. In this case was why existed other faces in the angle the camera, for the plates the results were precise, but in some sceneries the plates with characters as the following B, O, I, D, Q, Z can be confused by 8, 0, 1, 0, 0, 2 respectively, in this case we had to adjust to local format and obtain the correct angles.

The technology implementation how is described above with patterns recognition, have infinite implementations in different contexts, from selection of fruits, objects, to current applications (The measurement of body temperature) validation in shopping center, airports, hotels of the faces for identification the passports check in, and other applications.

References

[1] D. Heras, "Clasificador de imágenes de frutas basado en inteligencia artificial", *Killkana Técnica*, vol. 1, no. 2, pp. 21-30, 2017. https://doi.org/10.26871/killkana_tecnica.v1i2.79

[2] SicTransCore Latinoamérica, “Sic TransCore Sistemas de Identificación y control vehicular”, 2019. [Online]. Available at https://www.sictranscore.com/

[3] V. M. Arévalo, J. González, G. Ambrosio, “La Librería de Visión Artificial Opencv Aplicación a la Docencia e Investigación”, 2004. [Online]. Available at http://mapir.isa.uma.es/varevalo/drafts/arevalo2004lva1.pdf

[4] N. Carvajal Ahumada, “Reconocimiento Fotográfico De Patentes”, thesis, Pontificia Universidad Católica De Valparaíso, Valparaíso, Chile, 2018.

[5] E. Guerra Monterroza, “Reconocimiento de primitivas 3D, usando autocorrelación y ANFIS”, *Visión electrónica*, vol. 1, no. 1, pp. 56-61, 2008. https://doi.org/10.14483/22484728.251

[6] J. C. Briñez de León, H. A. Fandiño Toro, A. Restrepo Martínez, J. W. Branch Bedoya, “Análisis de resolución en imágenes de fotoelasticidad: caso carga dinámica”, *Visión Electrónica*, vol. 11, no. 1, pp. 69–75, 2017. https://doi.org/10.14483/22484728.12789

[7] R. Jiménez Moreno, J. Martínez Baquero, L. Rodríguez Umaña, “Sistema automático de clasificación de peces”, *Visión electrónica*, vol. 12, no. 2, pp. 258-264, 2018. https://doi.org/10.14483/22484728.14265