Global Journal of Researches in Engineering: B
Automotive Engineering
Volume 22 Issue 1 Version 1.0 Year 2022
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 2249-4596 & Print ISSN: 0975-5861

Artificial Vision Prototype for People and Vehicle Characteristics Recognition

By Fanny Suárez Mosquera, Cristian Dangelis Ballén Martínez
& Gerardo Alberto Castang Montiel

Universidad Distrital Francisco José de Caldas

Abstract- In a world increasingly connected with technology and with the growing need for security in all aspects of life, and the need of security in the transport of child in educational establishments is a great challenge. In other areas, the need of security in areas such as hospitals, parking lots, securities transporters, airports, etc. This solution aims to recognize patterns (vehicle plates) and characteristics (facial patterns); one of the main premises of the prototype was the use of free software to create a successful, low cost and easy-to-use solution.

Keywords: telematicprototype, patterns recognition, arduino, artificial vision, free software.

GJRE-B Classification: DDC Code: 387.7 LCC Code: HE9765

Artificial Vision Prototype for People and Vehicle Characteristics Recognition

Fanny Suárez Mosquera [α], Cristian Dangelis Ballén Martínez [σ] & Gerardo Alberto Castang Montiel [ρ]

Abstract- In a world increasingly connected with technology and with the growing need for security in all aspects of life, and the need of security in the transport of child in educational establishments is a great challenge. In other areas, the need of security in areas such as hospitals, parking lots, securities transporters, airports, etc. This solution aims to recognize patterns (vehicle plates) and characteristics (facial patterns); one of the main premises of the prototype was the use of free software to create a successful, low cost and easy-to-use solution.

Keywords: telematicprototype, patterns recognition, arduino, artificial vision, free software.

I. Introduction

The prototype was create to provide an accurate and easy solution in security. With this prototype, different schools can know about different characteristics of drivers and vehicles that provide school route service, providing a validation tool that can be integrate with the mobility registration system. A recognition of facial patterns and features can be made all of this supported by *free software recognition libraries*, through 128-point feature vectors which makes face representation for comparison. Allowing it to be easy to use and a low cost solution it will be apply in a number of scenarios making it scalable. Enhancing the centralization of data by using a distributed database generating availability and a higher security level.

II. Analysis Stage

An analysis of the chosen hardware and software architecture is carry out, and the scope and limitations of this are define for subsequent implementation. The main objective is design an artificial vision prototype for facial recognition of drivers and school route plates for entry and/or exit from the institution.

III. Design stage

The recognition of the involved agents with the system is developed and the interaction between these with the prototype. Besides relating the model of hardware and software of the system, it is divide in the Software component and Hardware component.

IV. Hardware Model

This item is scalable and configurable since inside this prototypeis necessary to build a hardware component, which would allow the driver to move into the institution when start the validation circuit of the prototype (IP camera and the rest of the services of facial recognition and registration).This component consist of a circuit with an Arduino card given the capacity of interaction that it can have with the system. It could be easily adapted to systems of sensors, automatic doors, registers and other access devices.

V. Software Model

The software make the *interaction* of the system with the final user. The system has components of graphic interface where the administration and the configuration modules will be visualize besides seeing the alerts with each one of the different entries that occur. The development with*free and open source technologies* such as *Python, Java, JSF, Java programming languages* were use with free distribution tools for its development such as *NetBeans, notepad ++, VisualStudioCode, Mysql database engine and Maria DB*.

VI. Database Model

The database modeling is an 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. The MARIA DB database engine was use with a GPL license in which the configuration is carry out in order to establish the distributed database.

VII. Sysytem Agents

User interactions were define for this system and the specifications may change according to the environment to be use with the Hardware and Software solution.

Author α σ ρ: Universidad Distrital Francisco José de Caldas, Facultad Tecnológica, Bogotá D.C. Colombia.
e-mails: fsuarezm@correo.udistrital.edu.co, cdballenm@correo.udistrital.edu.co, gacastangm@udistrital.edu.co

Table 1: Definition of System Agents.

Agent	Description
Admin	Responsible for the administration of alerts, personnel and vehicles authorized to enter to the institution.
Watchman	Agent who receives notifications of unauthorized 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.

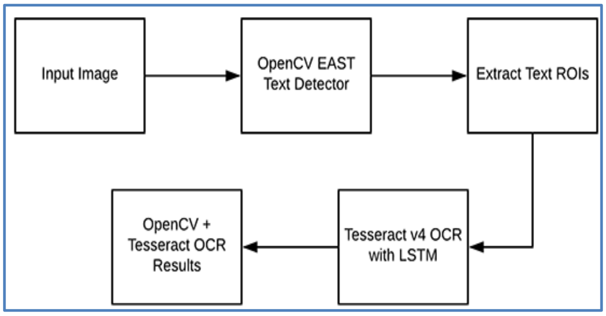
Source: authors

VIII. Architecture

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 or libraries and thus achieve the main objective to have a functional prototype ready for implementation. The used libraries and protocols are OpenALPR, OpenCV and RSTP. *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. *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 facial recognition, camera calibration, stereo vision and robotic vision. *RSTP* (Real Time Streaming Protocol) is a non-connection-oriented protocol. 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.

IX. License Plate Recognition with OpenALPR

We use OpenALPR library, which is an open source library that helps the automatic recognition of license plates. It operates for the video stream as follows: The image flow will be constantly extract from the IP camera in MJPEG format through RTSP on a video format, and OpenALPR will start the validation. The agent will start the process automatically and the library processes the flow as fast as possible while searching for plate images. When the plates are detect, the information will be write to a local beanstalk queue as JSON data. The library save the image in a configurable location as a jpeg image and run a separate process that empty the beanstalk queue upload the data to the server via HTTP. This library works with OpenCV for character recognition.

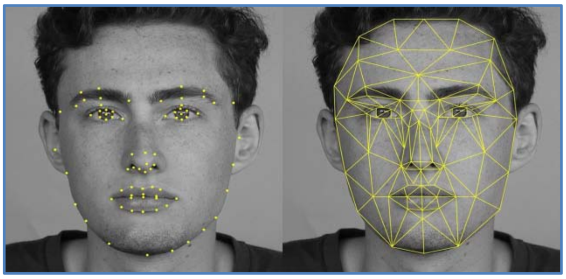

Input Image
OpenCV EAST Text Detector
Extract Text ROIs
Tesseract v4 OCR with LSTM
OpenCV + Tesseract OCR Results

source: https://www.pyimagesearch.com/2018/09/17/

Figure 1: Pipeline for Opencv

X. Face Recognition

This library built in python from the *Dlib toolkit* made in C++ contains deep learning algorithms, which allows to model abstractions providing accurate results reaching good accuracy model. 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 due to its impossibility of detecting profiled faces or low occlusion. Others based on deep metric learning rely on 128-point characteristic vectors making a representation of the face are used in this prototype. The algorithms can be accurate with the right distance threshold between points.

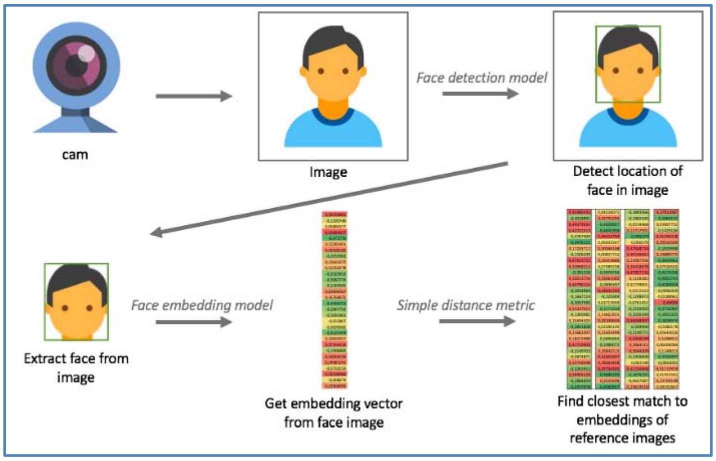


Source: https://encryptedtbn0.gstatic.com/images?q=tbn:ANd 9GcS0M

Figure 2: Dotted vector

The following steps identify the one face: The face detection model identifies the face's location inside the image. The embedding fed modelis use to obtain a vector of facial characteristics of size 128 points. It compares this vector with those of its "friends" and finds the most similar.

.

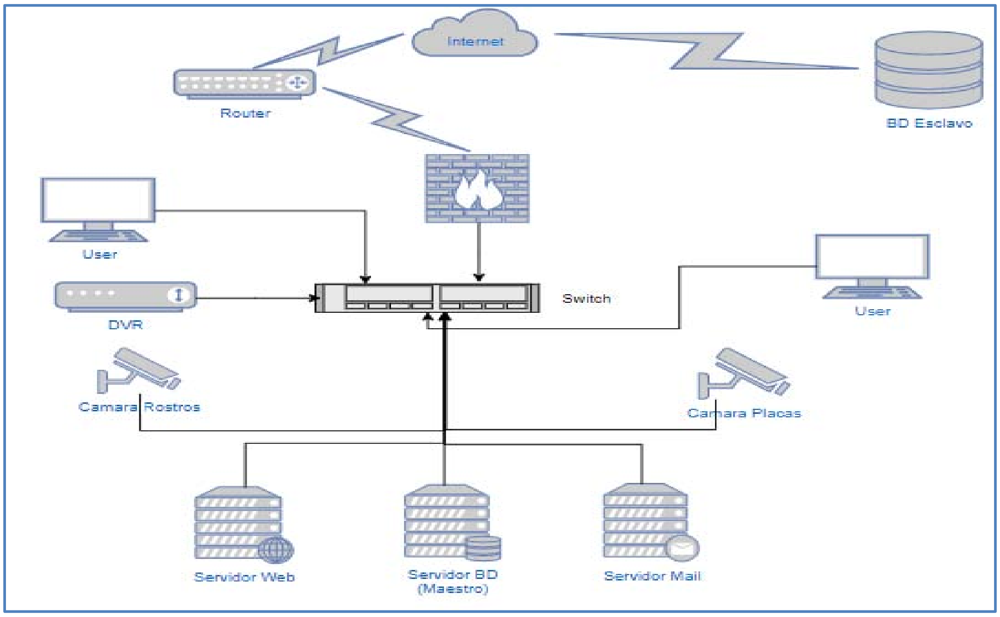

cam
Image
Face detection model
Detect location of face in image
Extract face from image
Face embedding model
Get embedding vector from face image
Simple distance metric
Find closest match to embeddings of reference images

*Source: https://miro.medium.com/max/700/1*R-ObQiGjDK4Njd5tSQEz5g.png*

Figure 3: Recognition cycle

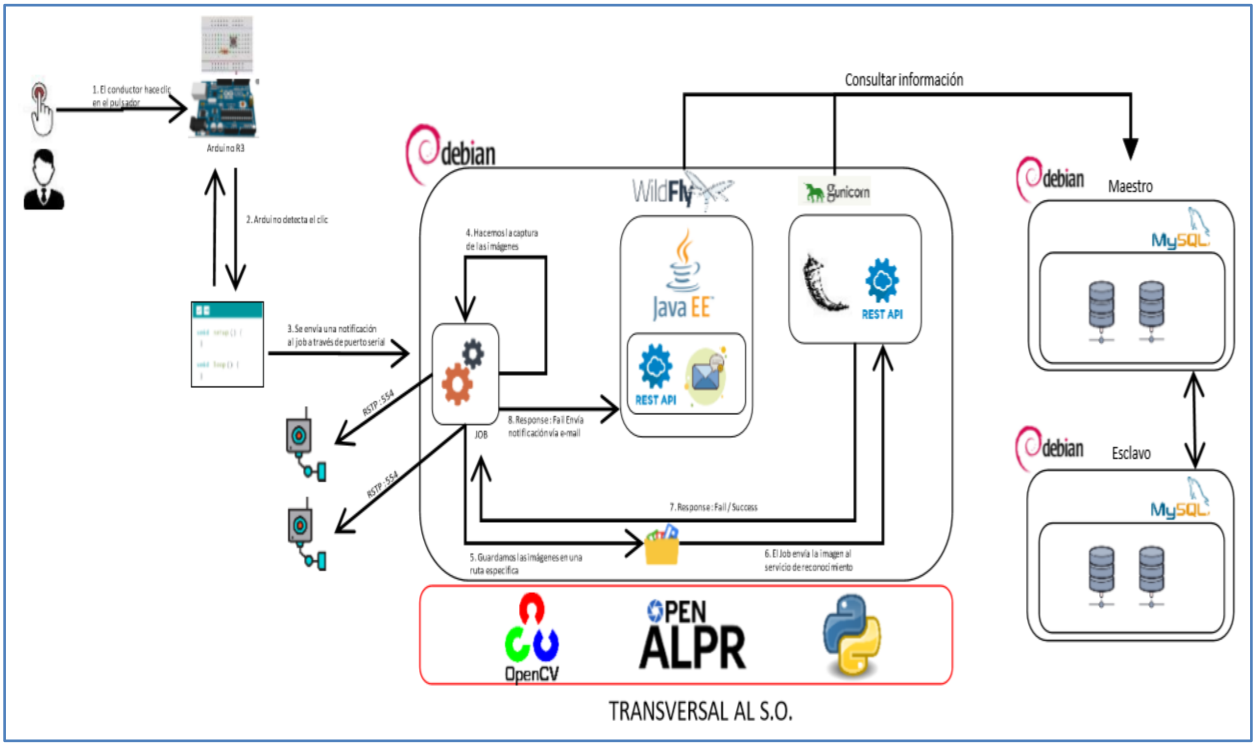
XI. Implementation Stage

The infrastructure defined for the system operation as shown in the below figure. Figure 4. Implementation of the prototype



Source: Authors

The prototype has components of hardware y software that interoperate in this form: *Arduino Wiring:* This code is stored in Arduino's microcontroller to start the request and begin the whole process. It can be replace by the access component that the application requires. *Python Code:* It is the backend of the application where the processing of the images and comparison of these are located. *Java Code:* It is the frontend of the system since it provides a friendly interface to the Administrator and Watchman actors that make up the system. To have a better view of the system operation the following diagram has been validate.


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
4. Hacemos la captura de las imágenes
RSTP : 554
RSTP : 554
JOB
5. Guardamos las imágenes en una ruta específica
6. El Job envía la imagen al servicio de reconocimiento
7. Response : Fail / Success
8. Response : Fail Envía notificación vía e-mail
debian
WildFly
Java EE
REST API
Gunicorn
REST API
Consultar información
debian
Maestro
MySQL
debian
Esclavo
MySQL
OpenCV
OPEN ALPR
TRANSVERSAL AL S.O.

Source: Authors

Figure 5: Prototype Overview

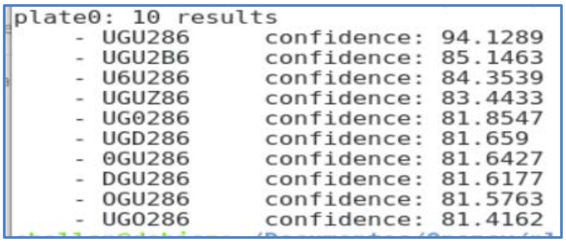
The diagram above describes the technologies used and the flow of information. An Arduino card and a push button is use to start the flow, which could be replaced by an access bar or a movement sensor. The Arduino board algorithm processes the message sent through the serial port and sends an alert to the JOB (Automatic Task). It takes a picture of the face and the license plate of the vehicle and stores it in a folder. The JOB (Scheduled Task) sends the 64-based images to the recognition service for analysis. The service generates a response indicating whether the recognition was successful or not to the JOB. 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.

XII. Hardware and Software Integration

We make the two components interact with each other and give functionality to the final product and the following elements were define for its operation: The Hardware model, Job and Web Services and Web Application (JEE).

XIII. Test Stage

The accuracy of a project is define by the test cycle aim to identify achievements and limitations obtained in the development of the prototype. For this is need a vehicle and a person who will carry out the entry simulation to the institution. When analyzing the image the results are:

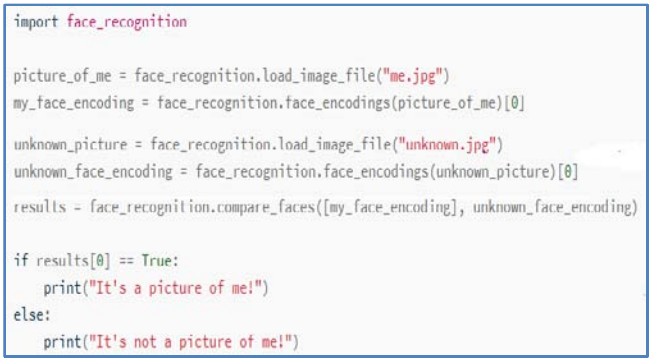


```
plate0: 10 results
    - UGU286        confidence: 94.1289
    - UGU2B6        confidence: 85.1463
    - U6U286        confidence: 84.3539
    - UGUZ86        confidence: 83.4433
    - UG0286        confidence: 81.8547
    - UGD286        confidence: 81.659
    - 0GU286        confidence: 81.6427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

Source: Authors

Figure 6: Vehicle Plate Identification

A list of ten possible plates with percentage of accuracy are see. We can obtain 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 intend to be identify and 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 to be analyze is the desired person. Some guidelines that must be take 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 take image for the identification. The validation algorithm returns a vector with the analysis response. If true value is return, means that it recognized a match with one of the base images.



```
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: https://github.com/openalpr/openalpr

Figure 7: Face Recognition Algorithm

XIV. Results Stage

The prototype's confidence levels may change with conditions of distance, comparative and luminosity. Recognition levels are high when the prototype has the right photo profiles for optimal comparison. In the plate case found that the prototype could be highly adaptive. The OCR not only recognizes the model of the plates that are hand as standard but also recognizes those parameterized in the system.

XV. Current Works

To have a more complete overview of the potential of the Open ALPR tool, which uses OpenCv, it is research and is evident that these libraries have a wide use from implementation of face recognition in major airports, and too be a solution for optimal fruit sorting with high quality standards.

XVI. Future Works

This application is highly scalable and modular to be use in other works where it can be integrate in a simple way. The version implements pattern recognition and we can will detect plates from various countries, color, vehicle brand and model; and the library for face recognition can be integrate easily.

XVII. Conclusions

The face recognition effectiveness ranges between 96% and 99% depending on the image type to be analyzed having a higher effectiveness if it has the three angles of the face which are the 128 point feature vectors for OpenCV works and which it makes the comparison.

The Optical Character Recognition (OCR) component has an effectiveness of approximately 94% that depends on factors such as brightness, image capture angle, camera resolution and that the plate does not have any kind of obstruction because there may be a poor character reading.

These two great components integrated to the software component allow a powerful tool with a high rate of effectiveness where it can be evidenced that with the available tools in its free versions can provide very useful solutions in all common life areas.

References Références Referencias

1. Adrian Rosebrock, OpenCV OCR and text recognition with Tesseract, Last Update: July 5,2021. Link: https://www.pyimage-search.com /2018/09/17/opencv-ocr-and-text-recognition-with-tesseract/
2. Curso de introducción a la visión artificial con opencv y python. 10 de enero de 2017, Link: https: //www.cacheme.org/curso-vision-artificial-opencv-python/
3. Rodríguez Guerrero R, Vanegas C, Castang G. (2020).Python a su alcance. EditorialUD, 2020. Páginas 100-120. ISBN 978-958-787-181-4.
4. Diego Heras, “Clasificador de imágenes de frutas basado en inteligencia artificial”, *Killkana Técnica,* Vol. 1, no. 2, pp. 21-30, 2017.
5. SicTransCore Latinoamérica, *Sic TransCore Sistemas de Identificación y control vehicular*, 2019. [Online]. Link: https://www.sictranscore.com/
6. V. M. Arévalo, J. González, G. Ambrosio; Opencv y su aplicación ala docencia einvestigación; Dep.Sis. y Aut. Universidad de Málaga, España. [Online]. Link :http://mapir.isa.uma.es/varevalo/drafts/arevalo 2004lva1.pdf
7. Bastián Nicolás Carvajal Ahumada, *Reconocimiento Fotográfico De Patentes*, Facultad de Ingeniería, Pontificia Universidad Católica De Valparaíso, Valparaíso, Ciudad de Chile, 2018. Link: http: //opac.pucv.cl/pucv_txt/Txt-4500/UC-C4791_ 01.pdf
8. R.C. Gonzalez, R. Woods, Digital Image Processing, 3a ed. USA: Prentice Hall, 2008.
9. M. Nixon, A. Aguado, Feature Extraction & Image Processing, 2a ed. Londres, UK: Elsevier Ltd., 2008.
10. D. Savakar, "Identification and Classification of Bulk Fruits Images using Artificial Neural Networks," Int. Journ. Eng. Innov. Tech. (IJEIT), vol. 1, no. 3, 2012.
11. M. Hassoun, Fundamentals of Artificial Neural Networks, MA, USA: A Bradford Book, MIT Press, 1995.