Passenger Flow Analysis and Fraud Detection System for Buses

João Correia
joao.p.correia@ist.utl.pt

Instituto Superior Técnico, Lisboa, Portugal
Setember 2020

Abstract
Currently, there is a major mobility problem in urban areas, due to the lack of efficiency of public transport due to the increasing concentration of the population and the great diversity of needs of this population. To make public transport more efficient, it is then necessary to understand what the mobility needs of passengers are, through an analysis of the route they take. On the other hand, there is a high rate of fraudulent user behavior, causing a negative impact on the accounts of public transport companies. The objective of this project is to evaluate these problems with the aid of computer vision, where a passenger counting system will be developed first, which analyzes the number of users entering and leaving each station / stop of the transport services in order to satisfy the first issue mentioned. Secondly, a system for verifying the passenger’s pass validation through facial recognition will also be developed, which analyzes whether it was in fact the pass holder who validated it. The evaluation made to the developed system showed a good performance in the facial recognition and fraud analysis module and a success rate in the count of inputs and outputs of 79 %. However, it was not possible to integrate the system developed with the ticketing system of the company Card4B due to the Covid-19 pandemic. No paragraph breaks.

Keywords: Public Transport; Computer Vision; Facial Recognition; Passenger Count

1. Introduction
The main purpose of this dissertation is the development of a system, using cameras and making image analysis, capable of satisfying two distinct objectives that aim to solve the problem of inefficiency in transport systems and the problem of the high percentage of fraud in validations passengers. The two distinct objectives are as follows:

- Identify the number of passengers entering and leaving a specific stop at a specific time as well as the total number of passengers in the vehicle in real time, in order to make this information available to other modules implemented by Card4B, so that it is possible to make a better management of resources of companies providing public transport services with the aim of improving the quality of these services.

- Analyze the existence of fraud in the validation of the passenger and make this information available to other modules implemented in Card4B so that it is possible to manage the possible penalties to be paid to the passenger who committed the fraud, such as the existence of a BlackList that contains the IDs of fraudulent users so that they are prevented from making new validations.

Both objectives developed in this dissertation are developed with the aid of computer vision, through the analysis of images captured by different cameras placed on the vehicle, the first objective being an analysis of the images identifying the number of entries and exits in real time and the second objective is facial recognition to identify the passenger.

2. Related Work
Before starting to develop the solution, it was important to better understand the problem to be solved as well as the existing solutions to these challenges. For this reason, a survey was made of several passenger counting systems and different solutions to perform facial recognition.

2.1. Passenger Count
One of the methods most used by public transport companies for counting passengers is the pass reader or ticket sale system, used for the ticketing function, but which ends up offering a sense of the number of tickets in the vehicle. This method has a disadvantage, since tickets or passes are sold in urban areas, that is, the passenger can travel freely within a geographic area, which does not allow checking the exact location of the passenger’s departure.

Other conventional systems for passenger counting are systems with infrared sensors, figure 1,
which can be active or passive, and systems with carpet sensors, figure 2, which are normally placed on the bus steps.

Figure 1: Infrared sensor of the active type on a bus [1]

Figure 2: Carpet sensor on a bus [1]

However, the systems that have the best efficiency are passenger counting systems through image analysis.

The article [4] presents a software solution for counting people through computer vision developed in python and with OpenCV.

This article, in addition to providing a description of how people counting algorithms work, also provides the source code. Passenger counting can be divided into two important steps, object detection and object tracking, the first of which is based on Deep Learning and is therefore computationally more expensive than the tracking phase.

When talking about object detection based on Deep Learning, there are three main types of methods, Faster R-CNNs, YOLO and SSD.

To create an object detection network, it is necessary to use an existing neuronal network architecture. A possible architecture is MobileNets which was designed for devices with limited resources, such as mobile devices.

It is possible to arrive at a fast and efficient method based on Deep Learning for object detection, combining the MobileNet architecture and the structure of the SSDs [6].

The solution developed in the article uses a pre-trained model in a data set COCO (Common Objects in Context) [5] being able to detect 20 types of objects in images such as airplanes, bicycles, birds, boats, bottles, buses, cars, cats, chairs, cows, dining tables, dogs, horses, motorcycles, people, pots with plants, sheep, sofas, trains and TV monitors.

After detecting the desired object, it is necessary to understand the direction in which it moves in order to count people.

Object tracking consists of making object detections, containing the coordinates of the object’s bounding box, creating an ID for each of the detections and tracking each of the detected objects as they move along the video. [7].

After obtaining the object’s bounding box, produced by an object detector, the center point of the bounding box is calculated, which is called Centroid, as shown in the figure 3.

Figure 3: Centroids [7]

2.2. Facial Recognition

This section aims to identify various types of existing solutions to realize the second proposed computer vision use case, facial recognition.

The best-known computer vision APIs were studied, such as the Google, Amazon, Microsoft, IBM and Kairos API to select a service to implement in the solution proposal.

Despite knowing what each API is capable of, it is important to compare the facial recognition performance of the different APIs.

In the article [2] a performance test was performed between the APIs of Amazon, Kairos and Microsoft where a set of facial recognition data from the university of Essex in the United Kingdom was used.
To perform this experiment, 50 different users were used and 5 images of each user were selected to train the API. For each user, 3 positive tests and 3 negative tests were performed, with the positive test consisting of 3 different images of the same user and the negative test 3 images of random users, with a total of 6 tests for each of the 50 users.

The API after having the training images is ready to receive the image to be tested, returning the confidence value of a match where it suggests the probability that the tested image corresponds to the user in question.

In order to analyze the performance, the True Positives and False Positives were recorded for scenarios of different levels of accuracy.

For a more accurate scenario, a cut confidence of 95% was used for the API of Amazon and Kairos and a cut confidence of 80% for the API of Microsoft, obtaining the results shown in table 1.

<table>
<thead>
<tr>
<th>API</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td>100%</td>
<td>99.3%</td>
</tr>
<tr>
<td>Microsoft</td>
<td>100%</td>
<td>87.3%</td>
</tr>
<tr>
<td>Kairos</td>
<td>100%</td>
<td>72%</td>
</tr>
</tbody>
</table>

Table 1: Resultados do teste com corte 95% para Amazon e Kairos e 80% para a Microsoft [2]

A free alternative to using an API would be to do facial recognition locally with the help of OpenCV.

In the article [3] an application was developed capable of performing facial recognition using OpenCV where it is possible to train the model with photos before performing facial recognition. The author of the article tested the system with only 2 people, using 6 learning photos for each person, where he obtained results of confidence values in the correspondence between 40% and 70%.

After studying several applications and services that allow to perform facial recognition, it was concluded that the API that presents better performance and easier to use is the Amazon Rekognition API according to the article [?], and has the advantage, in relation to the OpenCV library, because the API can obtain a certain facial recognition correspondence using only one photo of the user for learning, which is not possible in OpenCV.

According to the article [3], OpenCV needs approximately 20 learning images per user to have matches with higher confidence values, and thus closer to the confidence values obtained with the Amazon API.

3. Architecture

The architecture of the solution is divided into two main components that aim to fulfill the two use cases, the passenger count and the fraud analysis through facial recognition, and for that reason two different modules were designed, the Passenger Counting Module and the Facial Recognition Module.

Both use cases are processed with the aid of computer vision, so there is a camera responsible for capturing faces to later perform a fraud analysis through facial recognition, and a camera on each bus door responsible for capturing images in order to count inputs and outputs. The position of the cameras is shown in the figure 4.

![Figure 4: Position of cameras on board the vehicle (1 - Rear door camera, 2 - Face recognition camera, 3 - Front door camera)](image_url)

Buses with the Card4B system are equipped with a Android Tablet that allows printing tickets and invoices, reading and validating cards and QR Codes, as well as making payments with a bank card, with the driver of the vehicle being the user of the interface. The application that allows these features is called App Driver.

The Passenger Counting Module is responsible for counting passengers and associating the number of passenger entrances and exits at a given stop at a given date and time. The module starts when a bus door is opened and begins to receive images from the camera responsible for this task. Image analysis is performed in real time using computer vision in order to provide the vehicle occupancy number in real time.

The Facial Recognition Module is the module responsible for associating a validation ID with a face captured by the camera responsible for this task, which is located above the pass validator. When a user is validated, AppDriver sends a validationID to the application, and the Facial Recognition Module saves the photo of the detected face and associates the name of the saved file and the validationID in the database, to be done later fraud analysis through facial recognition. This analysis is not performed in this module, but on the external server, because a facial recognition API is used and an Internet connection is required.

The Server is located in the garage, where the vehicle ends its route, so that data can be trans-
ferred between the bus and the server via Wi-Fi. The main role of the server is to centralize the information collected by the vehicles and check the validations made on the buses through facial recognition. The Amazon Rekognition API is used to perform facial recognition. After performing the fraud analysis, the server informs the BackOffice Ticketing Module implemented by Card4B if the validation was well designed or if it was a fraud.

4. Implementation
In order to meet the objectives of the dissertation in relation to passenger counting and fraud analysis, an application was developed to run in Android environment that aggregates the Passenger Counting Module and the Facial Recognition Module.

This application was developed with the Kivy library, which allows the development of mobile applications in the Python language. A Python application was also developed to run on the server in order to perform fraud analysis and store the data collected by the first application.

The Passenger Counting Module consists of an adaptation of the [4] project analyzed in the section 2, it consists of a people counting program through visual computation based on deep learning, developed in python with the OpenCV library.

Object detection is performed using a deep learning object detection method, the SSD (Single Shot Detectors), on a MobileNets network that is designed for devices with limited resources.

The Facial Recognition Module is responsible for associating a validation ID with a face captured by the camera responsible for this task, which is located above the pass validator.

This module starts to work as soon as the application starts running, receives images from the camera and makes a face detection in real time with the help of the OpenCV library.

This face detection returns the coordinates of the bounding box of the face, which allows, when capturing the photo, to save only the face image in order to save space on the device, reducing the size of the saved file.

At the time of user validation, AppDriver sends a validationID to the application. The photo of the detected face is saved and the name of the saved file is associated with a validationID in the database.

The main objective of the server is to centralize the information collected by the vehicles and perform the fraud analysis through facial recognition. The server is a machine that runs a program developed in Python and has an Internet connection in order to make requests to the Amazon API and exchange information with other Card4B modules.

On the server it is also possible to register new users in the system, also registering the user’s face in the Amazon API’s facial collection to enable fraud analysis through facial recognition.

To perform the fraud analysis, the server at each entry in the validation table, sends the respective photo to the Amazon Rekognition API to perform facial recognition.

The API receives the photo and analyzes the detected face, extracting the facial features in order to find a match with some face registered in the API’s face collection.

When a match is detected, the API returns the corresponding face ID, as well as a confidence level value for that match.

When no match is detected, that is, when the match confidence level is less than 70%, the API returns an empty list of matched faces. If the API returns the empty list, it means that there was fraud, as the individual who performed the validation is not registered.

Another case of fraud is when the detected individual is registered but validated with a pass that does not belong to him. In order to detect this fraud, a comparison is made between the pass ID detected at the time of validation and the pass ID that corresponds to the Face ID returned by the API in the database containing the data for each user. If the IDs are the same it means that the validation is valid, otherwise the fraud was detected.

5. Results
Due to the Covid-19 pandemic it was not possible to test in a real environment, being tested at home, trying to simulate a real environment.

5.1. Passenger Count
In order to be able to evaluate the passenger counting system without carrying out the implementation on the buses, two different types of tests were performed. In the first test, example videos of people passing on the street and entering buses were used, and in the second, the AXIS M1045-LW camera was assembled in order to simulate a bus entrance.

In these tests, true and false passages were analyzed as well as undetected passages.

In the test with the two videos of people passing on the street, figure 5, the system obtained an accuracy of 100% in the first video and 93.3% in the second.

10 videos of people getting on and off buses were used in the second test, figure 6, in order to have a notion more similar to the ideally tested environment.

When analyzing the results obtained in the second test, it is possible to verify that the counting system did not perform as positively as in the first test.
It is also possible to see that the system has a greater accuracy in counting inputs than in counting outputs, having an average accuracy in counting inputs of 47.63 % and an average accuracy in counting outputs of 28.54 %.

To perform the count in real time, the AXIS M1045-LW camera was mounted, since it was not possible to obtain the AXIS M3044-WV camera. The camera was mounted at a height of approximately 2 meters, with an inclination of approximately 45 degrees.

The purpose of this test is to simulate a bus entrance, which is why several "entrances" and "exits" were made in the direction of the camera, bypassing a virtual line that symbolizes the entry of the vehicle. Several tests were carried out in different conditions (with light, in the dark, with just one person, with several people and finally with one person with suitcases) in order to understand the system’s performance.

After analyzing the results, it was possible to notice that the system has some difficulty in counting entries or exits when passengers are very close together, since tests with only one person are relatively better than with two or three people.

It was also possible to notice that the system has a better performance counting entries than exits, since when entering the passenger has his face turned towards the camera, which facilitates the detection of people, which does not happen when he is leave because he is facing away from the camera.

The average success rate for counting entries is 85.83%, and the average success rate for counting exits is 70.7%.

The average success rate of the system in counting inputs and outputs is 79 %.

5.2. **Face Detection**

The AXIS M1045-LW camera was used to evaluate the performance of the face detection algorithm.

Six people were tested in 6 different conditions, with nothing on their faces, with a hat, with sunglasses, with a mask, with all these accessories and in the dark.

The developed system was able to detect the faces without problem when the user was with nothing on the face, with the hat, with the sunglasses or in the dark. When the user used a mask, the system had some difficulty finding the face, however it was possible to capture the face with a mask from all the tested users.

The system was only able to detect the face with all these accessories in two users.

5.3. **Facial Recognition**

To evaluate the performance of facial recognition used in the developed system, photos of the faces detected in the previous test were used.

The 6 tested users were also registered in the system, where previously a photo of their face, captured by a Smartphone, was used to register the face in the API.

The Amazon Rekognition API was able to recognize all users when they were with nothing on their faces, having an average confidence value in correspondences of 98.70%. The same happened when
the same users had a hat, with an average confidence value of 96.76%.

The system was also able to recognize all users when they had sunglasses with the exception of one user, obtaining an average confidence level of 86.20%.

In the test with masks, the system only failed to recognize a user, obtaining an average value of 73.83%. Could not recognize any face in the test with all accessories. In the dark, the API was able to recognize all faces with an average confidence value of 95.70%.

It was then possible to obtain an average confidence value per user of 90.24%, disregarding the test with all accessories, since only two photos were tested.

5.4. Transfer Bus to Server
In order to have a small idea of the time that the bus has to be in the garage where the server is, in order to send all the information collected during the trips, the transfer time of images and databases of the system developed to the server was measured.

38 images of passenger faces and a database were transferred from the application to the server. The database contains a table with information on the 38 validations and contains a table with information on the 38 passenger entrances and 38 exits.

In order to analyze the transfer time, a network capture was performed using the WireShark program.

Between 200 and 1500 people per day can get on and off a bus, depending on the bus and the routes it takes. An approximation of the time that the bus has to wait connected to the server network has been calculated, based on the time calculated in the test with 38 passengers. The calculated result is shown in the table 2.

<table>
<thead>
<tr>
<th>Number of Passengers</th>
<th>Time (mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>00:04</td>
</tr>
<tr>
<td>200</td>
<td>00:21</td>
</tr>
<tr>
<td>1000</td>
<td>01:47</td>
</tr>
<tr>
<td>1500</td>
<td>02:41</td>
</tr>
</tbody>
</table>

Table 2: Data transfer time between bus and server

6. Conclusions
The main objective was a need for the Card4B company to have a system for counting and analyzing the passenger route and a fraud detection system through image analysis.

The work consisted in the development of an application capable of analyzing the number of passenger entrances and exits at a given stop through image analysis using real-time computer vision, being also able to collect the passenger’s face in the act of validating the pass, way to associate the face with the validated pass ID.

Due to the conditions imposed by the Covid-19 pandemic, it was not possible to integrate the system developed with the system of the company Card4B in time, and in turn to carry out tests in a real environment, that is, on a bus equipped with this system.

For this reason, the necessary Card4B modules were simulated in the implementation of the system.

The system is capable of counting passengers both in a bright environment and in full darkness, however it has some difficulties in detecting several passengers when they are very close physically, which may mean some lack of precision when a bus has a high occupancy rate. An average success rate of 79% was obtained in counting inputs and outputs in tests performed on the system.

Regarding facial recognition, the system performed well.

There are aspects to be improved and finalized in the system, the integration of the system developed in Card4B ticketing system being the most priority aspect.

References