electronics

Article

Traffic Signal Control System Based on Intelligent Transportation System and Reinforcement Learning

Julián Hurtado-Gómez [1], Juan David Romo [1], Ricardo Salazar-Cabrera [1,*], Álvaro Pachón de la Cruz [2] and Juan Manuel Madrid Molina [2]

[1] Telematics Engineering Research Group (GIT), Telematics Department, Universidad del Cauca, Popayán 190003, Colombia; julianhurtado@unicauca.edu.co (J.H.-G.); rtjuan@unicauca.edu.co (J.D.R.)
[2] Information Technology and Telecommunications Research Group (I2T), ICT Department, Universidad Icesi, Cali 760031, Colombia; alvaro@icesi.edu.co (Á.P.d.l.C.); jmadrid@icesi.edu.co (J.M.M.M.)
* Correspondence: ricardosalazarc@unicauca.edu.co; Tel.: +57-313-586-0304

Abstract: Traffic congestion has several causes, including insufficient road capacity, unrestricted demand and improper scheduling of traffic signal phases. A great variety of efforts have been made to properly program such phases. Some of them are based on traditional transportation assumptions, and others are adaptive, allowing the system to learn the control law (signal program) from data obtained from different sources. Reinforcement Learning (RL) is a technique commonly used in previous research. However, properly determining the states and the reward is key to obtain good results and to have a real chance to implement it. This paper proposes and implements a traffic signal control system (TSCS), detailing its development stages: (a) Intelligent Transportation System (ITS) architecture design for the TSCS; (b) design and development of a system prototype, including an RL algorithm to minimize the vehicle queue at intersections, and detection and calculation of such queues by adapting a computer vision algorithm; and (c) design and development of system tests to validate operation of the algorithms and the system prototype. Results include the development of the tests for each module (vehicle queue measurement and RL algorithm) and real-time integration tests. Finally, the article presents a system simulation in the context of a medium-sized city in a developing country, showing that the proposed system allowed reduction of vehicle queues by 29%, of waiting time by 50%, and of lost time by 50%, when compared to fixed phase times in traffic signals.

Keywords: traffic signals control; reinforcement learning; machine learning; intelligent transportation systems; vehicle queue

check for updates

Citation: Hurtado-Gómez, J.; Romo, J.D.; Salazar-Cabrera, R.; Pachón de la Cruz, Á.; Madrid Molina, J.M. Traffic Signal Control System Based on Intelligent Transportation System and Reinforcement Learning. *Electronics* **2021**, *10*, 2363. https://doi.org/10.3390/electronics10192363

Academic Editor: Beatriz L. Boada

Received: 30 August 2021
Accepted: 24 September 2021
Published: 28 September 2021

1. Introduction

High vehicle congestion causes around 40% of the world's population to spend at least one hour on the road every day [1]. According to a report published by the INRIX organization, drivers in Bogotá (Colombia), the city, spend the most time in traffic congestion worldwide [2], losing an average of 191 h in year 2019 due to traffic congestion.

Facts such as this underscore the importance of looking for solutions to these traffic problems, especially in developing countries such as Colombia. These problems can have different causes, such as insufficient infrastructure capacity, unrestricted demand, and lack of synchronization in traffic signals [3].

Traffic signal control is an important and challenging real-world problem, which aims to minimize the travel time of vehicles by coordinating their movements at road intersections [4–8].

Many cities in developing countries feature traditional traffic signal control systems (TSCS), programmed with fixed times calculated by assumptions of expected traffic. Changes to those fixed times are burdensome and can take a long time (several days or weeks) when the need for such a change is detected. Other cities feature systems with dynamic time assignments by using loop sensors that detect passing vehicles and help determine the direction of an intersection requiring a longer green light time to decrease traffic queues [5]. Usage of loop sensors for dynamic allocation of times of traffic signal plans has some limitations, mainly in heavy traffic where there are many stopped vehicles that do not pass through the sensors [4,5].

New data sources (such as GPS equipped vehicles, navigational systems, and traffic surveillance cameras) are quickly becoming available and can be used as inputs for traffic signal control purposes [4–6]. In recent years, new artificial intelligence techniques have tried to take advantage of such data sources to improve operation of traffic signals. Such techniques include fuzzy logic algorithms [9,10], swarm intelligence [11,12], and Reinforcement Learning (RL) [4,13–19].

RL is the most used traditional Machine Learning approach for traffic signal control because it does not need an exact model of the traffic flow behavior in a network of intersections. RL is a concept where a system can learn how to bring the underlying process in a desired state according to the defined reward function using available actions. The system learns an optimal policy how to do that, and in the case of traffic signal control, the optimal signal program is learned. An RL-based agent gains knowledge through its learning process [15].

RL techniques are used to improve traffic signal control through trial-and-error searches. Compared with traditional transportation methods, RL avoids making hard assumptions about the traffic and learns directly from the feedback by trying different strategies [4].

There are four main components to formulate a traffic signal control problem under the framework of RL [5]:

- Reward. A reward defines the goal in a RL problem.
- State. The state captures the situation on the road and converts it to values.
- Selection of action scheme. Different action schemes also influence the performance of traffic signal control strategies.
- Coordination strategy. Achieving coordination is one of the challenges complicating the signal control problem.

The Q-learning algorithm is one of the most used RL approaches for traffic control [20] because it is relatively simple when compared to other types of RL algorithms. The Q-learning algorithm uses a table (Q-table), which stores values known as Q-values, that map to a state-action combination. A Q-value for a state-action combination represents the quality of an action taken from that state. Better Q-values imply better chances of getting greater rewards [15].

The definition of the state and the reward for applying RL to traffic signal control show is a key issue. Regarding the reward, it is important to consider the ultimate objective in urban transportation: minimizing vehicle travel time [4,5]. Travel time is influenced not only by traffic signals, but by other elements such as queue length, waiting time, speed, throughput, and delay. Each one of these elements could be considered to define an adequate reward. Previous studies use combinations of these elements (such as [13–17]) to try to establish an adequate reward. Regarding the state, elements such as queue length, waiting time, volume, delay, speed, phase duration, and congestion are considered to represent the states in traffic signal control [4,5]. Some proposals combine two or more of such elements or use novel elements such as images to represent the states, which results in a state with high dimensionality representation. In such cases, state approximation techniques such as tile- and coarse-coding, neural networks, etc., are considered.

Selection of some combinations of reward and states for application of RL increases the complexity of the proposed systems, making the dimensions of the matrices used by the RL algorithms too large and increasing the training time. In addition, improvement is not significant when compared to simple selection schemes using one element for reward and one for states [4,5].

A possible solution to this problem of large-dimension matrices (curse of dimensionality problem [21]), is the use of Deep Reinforcement Learning (DRL), which integrates deep learning methodologies with RL [21,22]. The main purpose of this integration is approximating the policy function to tackle the curse of dimensionality problem, by using Deep Neural Network (DNN) based models [15].

The approach used in this research, as an initiative of the authors to consider the system application context, is the development of a simple TSCS which can be implemented at intersections in cities of developing countries. The proposed control system does not use a very large Q-table, thus avoiding the dimensionality problem. Moreover, basic parameters were selected to achieve a simple design of both state and reward to avoid using DRL. The proposed system uses the Q-learning algorithm, and considers the number of vehicles on each approaching lane and the green light time of the traffic signals of each one of the four directions of the intersection (West-East WE, North-South NS, East-West EW, South-North SN) as the state; the selected reward is the vehicle queue length. The actions considered by the system are to leave the green light time unchanged, increase or decrease it for each one of the directions of the intersection. This simple implementation eases data collection and reduces training time.

In addition to the RL approach, it is important to consider the advantages of Intelligent Transportation Systems (ITS) in the traffic control system. An ITS plays an important role in public transport management, traffic control, security, and other issues. Traffic flow detection is an important part of an ITS. Based on the real-time acquisition of urban road traffic flow information, an ITS provides intelligent guidance for relieving traffic jams and reducing environmental pollution [23].

A case study was used to evaluate the operation of the proposed TSCS, at a traffic signal intersection in the city of Popayán, Colombia. To test the operation of the TSCS in the integrated system (mainly the correct operation of the recommendation times of the phases of the traffic signals), it was also necessary to develop a simulated prototype of the system using the SUMO software 1.7.0 version [24] because a live operational test in the actual infrastructure was not possible. The obtained results were compared and discussed, with the support of statistical graphs and comparative tables to facilitate analysis.

The remainder of this paper is organized as follows: Section 2 presents the materials and methods used; including the literature review, design, development, testing, and simulation phases; Section 3 presents the results of the TSCS testing and simulations; Section 4 discusses the results; and Section 5 presents the conclusions of the study.

2. Materials and Methods

This section presents the materials and methods used for each of the stages of the research (systematic literature review, determination of an adequate ITS architecture, design and development of a system prototype, and designing and developing of system tests).

2.1. Systematic Literature Review

It is also important to highlight that this work conducts research on "How to perform adequate traffic control at intersections with traffic signals, in cities with infrastructure and budget limitations, considering the necessary standardization of mobility services for adequate interoperability". The hypothesis raised to answer the research question establishes that "It is possible to do so by developing a TSCS, focused on recommending times for traffic signals, using Machine Learning, and based on ITS services".

The work described in this paper included a rigorous systematic literature review performed with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) methodology. In the literature review, the four stages of the methodology were performed (identification, screening, eligibility, and inclusion). Three databases were included in the identification phase: EBSCO, IEEE, and Science Direct. The query string used was: "Traffic control OR traffic lights OR ((intelligent transportation system) AND traffic)) AND artificial intelligence AND time", obtaining 3295 documents. After filtering for year and type of publication, a total of 1780 documents were obtained. In the screening phase, duplicate documents were eliminated and filtered by title and/or abstract, yielding 46 documents. In the eligibility phase, through specific criteria and review of the full document, 26 documents were selected. Finally, in the inclusion phase, the following four groups of documents were identified for which a qualitative and quantitative synthesis was performed:

- Services or systems for intelligent control of traffic signal time.
- Design or implementation of mobility services through ITS that are not directly related to TSCS.
- Methodologies, models, analyses, algorithms, devices, and studies related to counting vehicles and control of traffic signal time.
- Artificial Intelligences (AI) techniques implemented in mobility services, which cannot be included in any of the previous groups.

This systematic review made it possible to identify the important aspects to consider for TSCS development. From the 26 documents selected in the eligibility phase, the documents for the state of art were selected. The state of art documents [14], refs. [25–44] made it possible to identify important aspects for the work described in this paper. Through a detailed review, it was possible to identify which documents contained relevant aspects, and which characteristics of our work had not been considered by most of the works reviewed. Table 1 presents a summary of the analysis of the state of the art.

The novelty of the work described in this paper was determined by the relevant aspects for the implementation of a TSCS in the target context, which have not been considered by most of the works reviewed in the state of the art. The novel factors identified were:

- The use of an ITS architecture as a basis for developing the prototype.
- Collecting a dataset for traffic analysis. The work proposed the collection of videos of the selected intersection as a case study to evaluate traffic flow at seven different time ranges per day.
- Using an algorithm to size vehicle queues. The proposed TSCS includes an algorithm for counting vehicles from the video signals captured at the intersection.
- The development of the RL algorithm for recommendation of traffic signal times in a way as simple as possible, with basic parameters and a short execution time. Most proposals found use RL with multi-agent and/or deep learning, which increases system complexity.
- Consideration of the target context, in aspects such as budget, lane characteristics and types of intersections. In cities similar to Popayán, traffic networks with lanes with similar flow of vehicles and with the same number of lanes are not common. Common intersections feature a main road or highway (with three or four vehicle lanes), and secondary roads with one or two lanes.
- Validation of traffic control modules in a real environment. This was possible due to the limitation in scope regarding the number of considered intersections. The proposed TSCS considers only an intersection, because the objective was to go beyond a simulation of the operation in a microscopic traffic software, considering its design, development, and implementation. As shown in Table 1, most of the reviewed references consider only the simulation of the system.

Table 1. Summary of evaluation of state of art.

Aspect/Work	[14]	[25]	[26]	[27]	[28]	[29]	[30]	[31]	[32]	[33]	[34]	[35]	[36]	[37]	[38]	[39]	[40]	[41]	[42]	[43]	[44]
Use of an ITS architecture					✔			✔		✔						✔					
Collecting a “dataset” for traffic analysis																					
Using a “dataset” for traffic analysis									✔		✔	✔									
Using an algorithm for recommendation of traffic light times	✔	✔	✔	✔	✔	✔	✔	✔			✔	✔	✔	✔		✔	✔	✔	✔	✔	✔
Using an algorithm to count vehicle queues								✔													
Validation of traffic control modules in a real environment					✔					✔			✔		✔	✔					
Complete system validation through simulation	✔	✔	✔	✔	✔	✔	✔	✔		✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
Context of a medium-sized city in a developing country	✔					✔								✔							

The proposed TSCS only considers the Q-learning algorithms, because one of the objectives of the work was to find an adequate traffic control option that would facilitate implementation in the considered context. Q-learning algorithms allow the implementation of RL without a steep increase of system complexity in terms of required components for information collection.

2.2. Determination of an Adequate ITS Architecture for the TSCS

An ITS aims to promote the safety and mobility of transport, reducing traffic and mitigating air pollution, among other benefits. These ITS are implemented in most developed countries through an ITS architecture and its services. An ITS architecture allows to identify the different components (modules) composing a certain service, describes the interaction between the actors (service users) and the components, and the information (content and structure) that will be exchanged between the components [45].

Reference architectures are used to define a context-aware suitable architecture for the TSCS. The most used reference architectures in the world include the American architecture, known as Architecture Reference for Cooperative and Intelligent Transportation (ARC-IT) [46]; the European architecture, known as European Intelligent Transport Systems Framework Architecture (FRAME) [47]; and the ISO 14813 architecture [48]. These ITS architectures are very well documented and updated.

ITS architectures evaluated in the Colombian environment were also reviewed. The ITS national architecture initiative for Colombia, based upon ARC-IT, was launched in 2010 [49]. The National ITS Architecture of Colombia is a guide for the ITS integration in Colombia. Although it was an appropriate initiative at the time, it did not achieve its mission and it has not been properly updated and adapted since its launch. Although the Colombian government has advanced in terms of ITS standardization (Act 1450 of 2011, Article 84; Decree 2060 of October 2015 [50]), this has been insufficient for adequate design, development, and implementation of mobility services in intermediate cities. After conducting a review of the different service sets of the ARC-IT, FRAME, and ISO architectures, several elements were chosen to build a suitable architecture for the TSCS.

In the ARC-IT architecture [46], the service area that best suits the TSCS is the Traffic Management (TM) area. After reviewing the services proposed in this area, the Traffic Signal Control (TM03) and Traffic Metering (TM05) services were considered because the system must measure the traffic flow at an intersection to determine the better times for their traffic signals, thus reducing the vehicle waiting time. Figures 1 and 2 show the architecture of the TM03 and TM05 services.

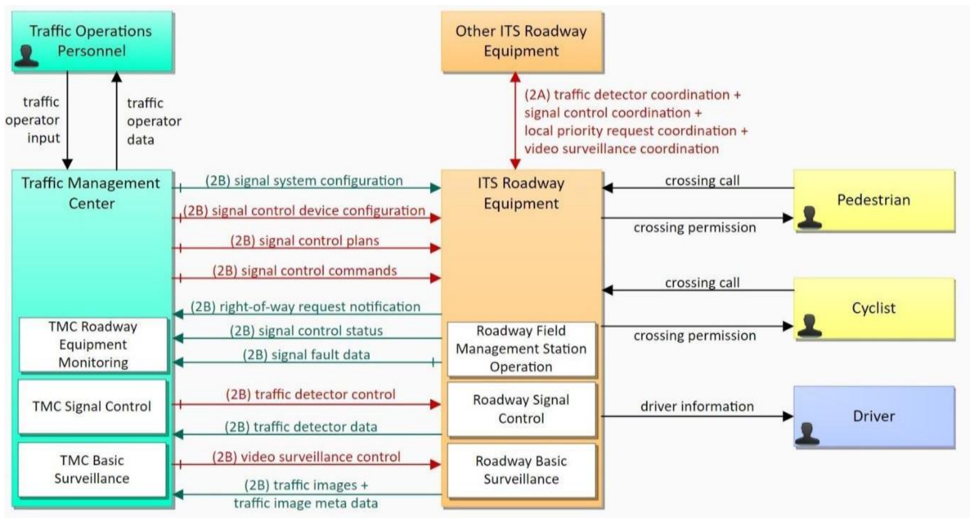

Traffic Operations Personnel
Other ITS Roadway Equipment
traffic operator input
traffic operator data
(2A) traffic detector coordination + signal control coordination + local priority request coordination + video surveillance coordination
Traffic Management Center
(2B) signal system configuration
(2B) signal control device configuration
(2B) signal control plans
(2B) signal control commands
(2B) right-of-way request notification
(2B) signal control status
(2B) signal fault data
(2B) traffic detector control
(2B) traffic detector data
(2B) video surveillance control
(2B) traffic images + traffic image meta data
ITS Roadway Equipment
TMC Roadway Equipment Monitoring
TMC Signal Control
TMC Basic Surveillance
Roadway Field Management Station Operation
Roadway Signal Control
Roadway Basic Surveillance
crossing call
crossing permission
Pedestrian
crossing call
crossing permission
Cyclist
driver information
Driver

Figure 1. ITS Architecture for the TM03 (Traffic Signal Control) service proposed by ARC-IT [46].

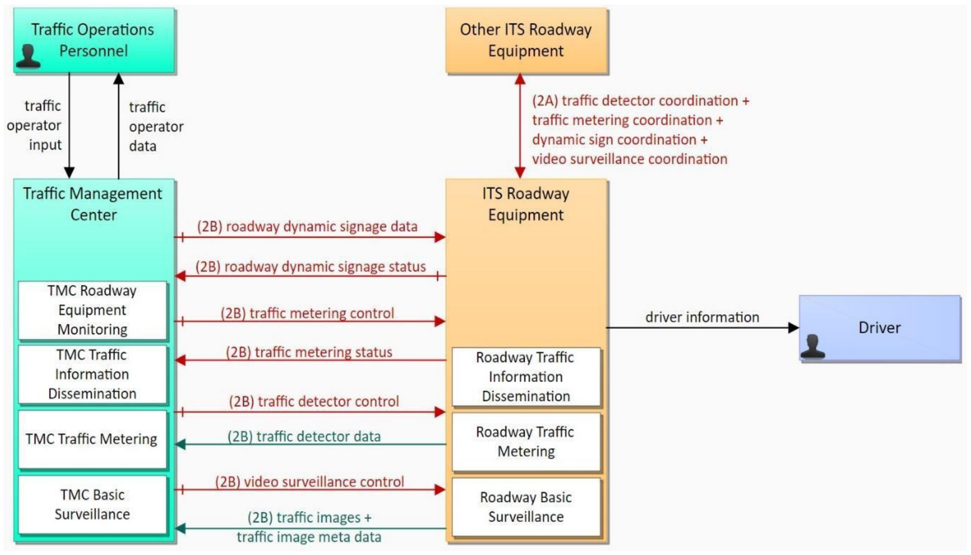

Traffic Operations Personnel
traffic operator input
traffic operator data
Other ITS Roadway Equipment
(2A) traffic detector coordination + traffic metering coordination + dynamic sign coordination + video surveillance coordination
Traffic Management Center
TMC Roadway Equipment Monitoring
TMC Traffic Information Dissemination
TMC Traffic Metering
TMC Basic Surveillance
(2B) roadway dynamic signage data
(2B) roadway dynamic signage status
(2B) traffic metering control
(2B) traffic metering status
(2B) traffic detector control
(2B) traffic detector data
(2B) video surveillance control
(2B) traffic images + traffic image meta data
ITS Roadway Equipment
Roadway Traffic Information Dissemination
Roadway Traffic Metering
Roadway Basic Surveillance
driver information
Driver

Figure 2. ITS Architecture for the TM05 (Traffic Metering) service proposed by ARC-IT [46].

The European architecture (FRAME) proposes ten service areas; area number three (Manage Traffic) is directly related to the TSCS. This area includes five high-level functions. The Provide Traffic Control function was chosen, because it provides facilities for traffic management using the road network, functionality to manage the urban parts of the network, and facilities that allow data collection on the use of the road network and prioritizing certain vehicles [51].

Figure 3 presents the proposed architecture for the TSCS, which considers results of the systematic review of literature, and the reviewed ITS reference architectures. Three of the components used in the TM03 and TM05 ARC-IT services are included: ITS Roadway Equipment, Traffic Management Center (TMC) and Vehicles (presented as Driver in ARC-IT). The traffic measurement, traffic signal control, and basic road surveillance modules are also included. The three components of the proposed ITS architecture allow traffic monitoring, data, and video collection, determination of the best traffic signal times for a given intersection, and programming of the traffic signals phases of any intersection.

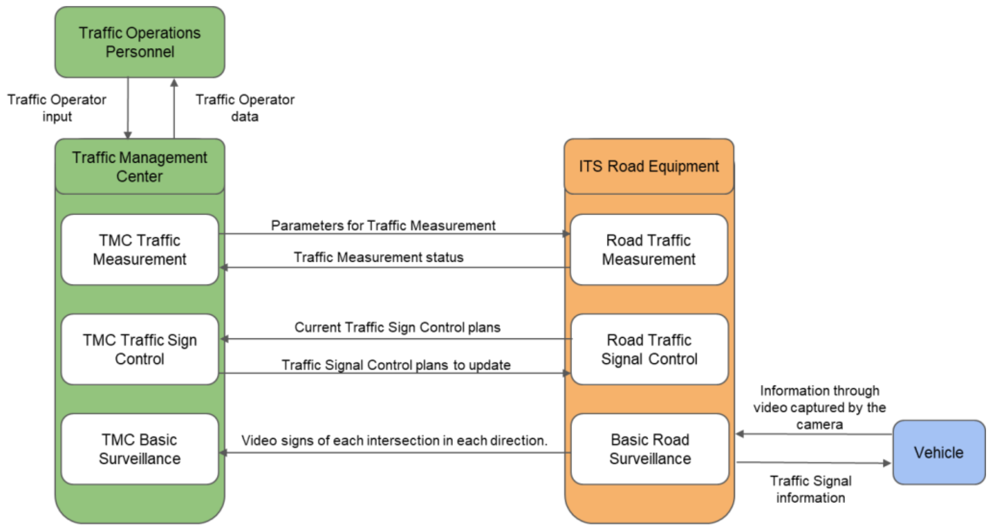

Traffic Operations Personnel
Traffic Operator input
Traffic Operator data
Traffic Management Center
TMC Traffic Measurement
TMC Traffic Sign Control
TMC Basic Surveillance
Parameters for Traffic Measurement
Traffic Measurement status
Current Traffic Sign Control plans
Traffic Signal Control plans to update
Video signs of each intersection in each direction.
ITS Road Equipment
Road Traffic Measurement
Road Traffic Signal Control
Basic Road Surveillance
Information through video captured by the camera
Traffic Signal information
Vehicle

Figure 3. ITS architecture proposed for TSCS.

The ITS traffic control system architecture proposed did not include the other actors (pedestrian and cyclist) suggested by ARC-IT architecture (Figures 1 and 2), because they have a very low degree of interaction with the traffic control system in the target context. The actors who want to cross the street near a high traffic intersection use elevated pedestrian bridges, or the pedestrian crossing zones located before the traffic signal, at the times that it is possible for them to cross.

The proposed ITS architecture uses high-level functions from the FRAME architecture described in the provide traffic control area (Provide Urban Traffic Management and Provide Traffic Predictions). The proposed architecture maintains most of the physical components proposed in ARC-IT, but also considers the functionality of the two mentioned reference architectures (FRAME and ARC-IT).

The "Basic Road Surveillance" functional module (white box in Figure 3) is responsible for capturing and relaying the video signals from each one of the directions of a given intersection to the TMC. This functionality allows the "Traffic Operations Personnel" to visualize the level of video traffic of a certain intersection with traffic signals.

The "Roadway Traffic Metering" functional module converts video signals into data to determine the length of the vehicle queue in each direction of the intersection. This data is also sent to the TMC.

Finally, the functional module defined as "Roadway Traffic Signal Control" oversees programming the appropriate phases of the traffic signals at a given intersection. Its input is the set of values that the TMC determines as adequate. The adequate values are calculated after analyzing the current traffic data at the intersection and the environmental conditions using an AI algorithm.

2.3. Design and Development of a System Prototype

Figure 4 presents a diagram of the prototype, with the selected components, modules and technologies.

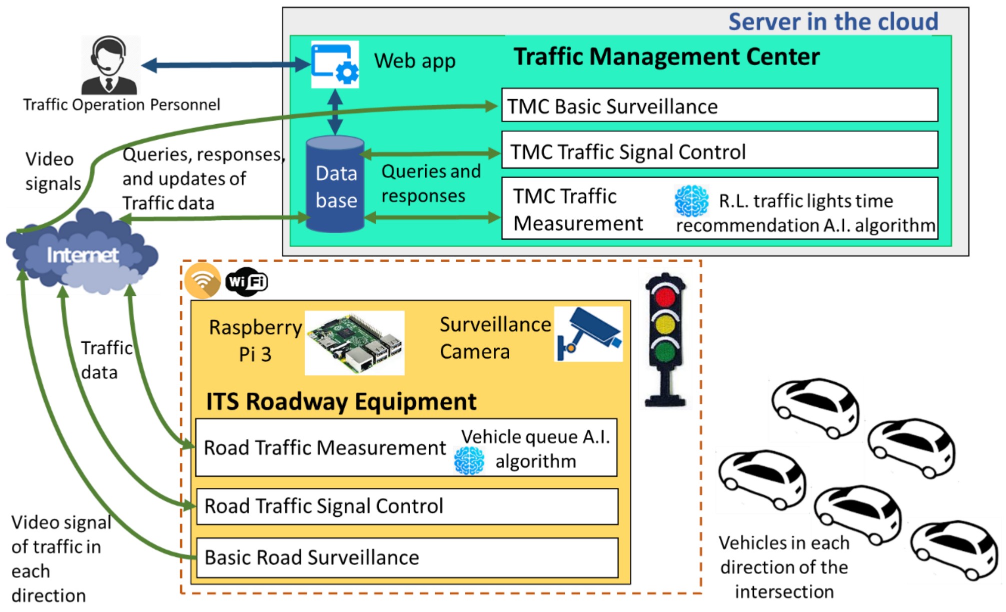

Server in the cloud
Web app
Traffic Management Center
Traffic Operation Personnel
TMC Basic Surveillance
TMC Traffic Signal Control
Video signals
Queries, responses, and updates of Traffic data
Data base
Queries and responses
TMC Traffic Measurement
R.L. traffic lights time recommendation A.I. algorithm
Internet
Raspberry Pi 3
Surveillance Camera
Traffic data
ITS Roadway Equipment
Road Traffic Measurement
Vehicle queue A.I. algorithm
Road Traffic Signal Control
Basic Road Surveillance
Video signal of traffic in each direction
Vehicles in each direction of the intersection

Figure 4. Proposed prototype for TSCS.

Next, in Section 2.3.1 the detailed design of the components and modules of the proposed prototype is presented. Section 2.3.2 presents the design and development of the RL traffic signal time recommendation AI algorithm, which is the principal smart component of TSCS. Finally, Section 2.3.3 presents the RL algorithm adjustments for simulation with the SUMO software are presented.

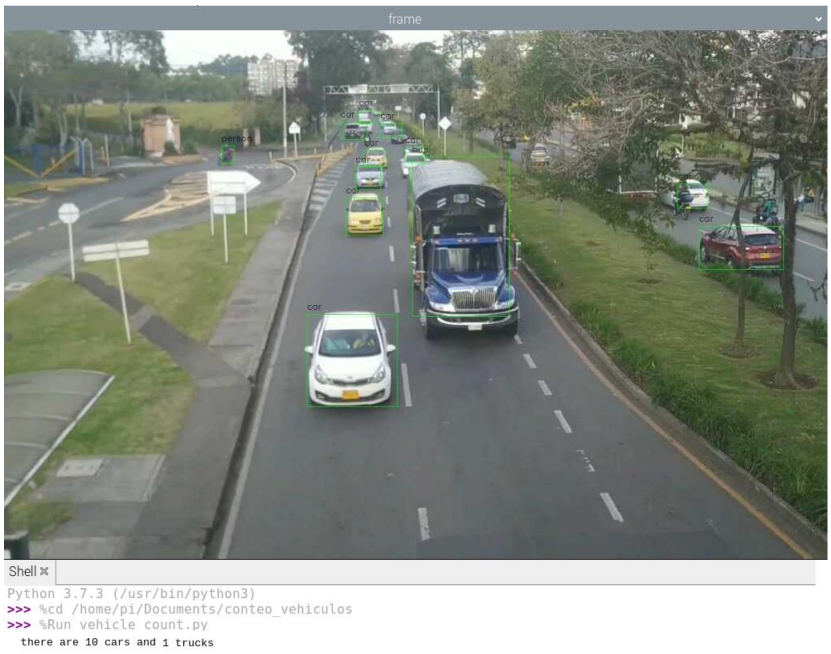
2.3.1. Description of Prototype's Components and Modules

Figure 4 shows the two physical components of the prototype:

1. ITS Roadway Equipment (ITS RE). Consists of a Raspberry PI 3 single-board computer (SBC) and a surveillance camera located in the structure of each of the traffic signals at the intersection. The Raspberry Pi SBC contains the three functional modules of the ITS RE: Basic Road Surveillance, Road Traffic Signal Control, and Road Traffic Measurement.
2. Traffic Management Center (TMC). The TMC is located on a server in the cloud. The three functional modules the TMC are: Basic Surveillance (TMC BS), Traffic Control (TMC TC), and Road Traffic Measurement (TMC RTM).

A description of the six functional modules of the prototype follows:

1. The Basic Road Surveillance (ITS RE BRS) module is responsible for capturing the image of the road at each traffic lane (with help of the surveillance camera) and transmitting it to the TMC. This module can capture the video signal from each one of the directions in the intersection (using a camera in each direction) to reduce costs.
2. The Road Traffic Signal Control (ITS RE RTSC) module sends the current traffic signal plan from the ITS RE to the TMC; this plan determines the order the intersection traffic signals change and the duration of the green light time of each of the signals. This information allows to determine the state of the intersection at the TMC. This module also receives updates of the traffic signal plan (from the TMC) to adjust them according to instructions issued by the Traffic Operation Personnel (TOP).
3. The Road Traffic Measurement (ITS RE RTM) module reviews video signals of current traffic in each direction of the intersection and counts the waiting vehicles (queue size), for this it captures an image periodically, detects and count the waiting vehicles using a computer vision algorithm. The used algorithm was an adaptation of the YOLO v3 algorithm [51]. An example of the vehicle counting performed by the algorithm is shown in Figure 5. YOLO version 3 is a real-time object detection system that, unlike other systems such as Region-based Convolutional Neural Networks (R-CNN), applies a single neural network to the entire image. This network divides the image into regions and generates bounding boxes and probabilities for each region, making this system 100 times faster than Fast R-CNN [52].



```
Python 3.7.3 (/usr/bin/python3)
>>> %cd /home/pi/Documents/conteo_vehiculos
>>> %Run vehicle count.py
  there are 10 cars and 1 trucks
```

Figure 5. Example of vehicle queue counting by algorithm (based in YOLO v3).

4. The TMC Basic Surveillance (TMC BS) module receives the video signals from each of the four directions of the intersection and presents them on the web app, with which the TOP interacts. The Flask Python library [53] and the Ngrok tunneling application [54] were used to allow simultaneous presentation of the four video signals on the Web application.
5. The TMC Traffic Control (TMC TC) module queries the traffic measurement parameters from the system database and transmits them to the RTM. Additionally, the TMC TC module queries the ITS RE about the current traffic state at the intersection. It requests the current state of the vehicle queue (short or long) and the green light time of each one of the directions of the intersection (short or long). The obtained response is an 8-bit array, where the four Most Significant Bits (MSB) are the green light times at each traffic signal (0 for short time, 1 for long time); and the four Least Significant Bits (LSB) represent the vehicle queue size in each direction (0 for short queue, 1 for long queue).
6. The TMC Traffic Measurement (TMC TM) module analyzes the current traffic state using the parameters obtained from the TMC TC module and makes a recommendation to update the time plan of the intersection's traffic signals. This module uses a RL traffic signal recommendation AI algorithm for such purposes. If the suggested recommendation differs from the current state, it is presented to the TOP in the Web app so that she can determine whether to update the traffic signal plan through the RTSC module. Information on the algorithm used in this module is detailed in Section 2.3.2.

A description of the data flows between the functional modules of the ITS RE and the TMC follows.

1. Traffic measurement parameters. This information is sent from the TMC TM module to the ITS RE RTM through a database request. The parameters define three aspects: the periodicity (in seconds) of traffic measurement in the ITS RE; the maximum size of the vehicle queue in each of the directions of the intersection; and the values (in seconds) of the short and long times of the green light in each of the intersection directions.
2. Traffic measurement status. Response message from the ITS RE RTM to the TM TMC. Contains the number of vehicles identified by the AI algorithm (queue sizes) at the intersection. This information is stored in the database.
3. Current traffic signal control plan. Information about the current traffic signal phases and cycle at the intersection is coded in a string (e.g., STSTLTLT that means Short Time, Short Time, Long Time, and Long Time for each direction of intersection), which is sent from the ITS RE RTSC module to the corresponding module in the TMC. The traffic signal phases include the order in which the traffic signals change and the green light times (in seconds) for each one of them; this information is also stored in the database.
4. Updates to the traffic signal control plan. Message sent from the TMC TSC module to the respective ITS RE module through the database to update the traffic signal phases based on the recommendations that have been determined using the AI algorithm. The message includes the order in which the traffic signals change and the green light times (in seconds) for each traffic signal coded into a string (e.g., LTLTSTST).
5. Video feeds of the intersection in each direction. This message includes video feeds (in low or medium resolution) of the intersection in each direction, so the Traffic Operations Personnel (TOP) can see the respective videos.

2.3.2. RL Traffic Signal Time Recommendation AI Algorithm

The RL process is explained in Figure 6. The figure identifies the mentioned elements (states, actions, reward, and environment). The agent initially reviews the system environment (current state and possible actions), then decides which action to take (among the possible ones defined in the environment), executes the action, and in response receives a reward and a new state. This process is repeated continuously until an optimal state is achieved.

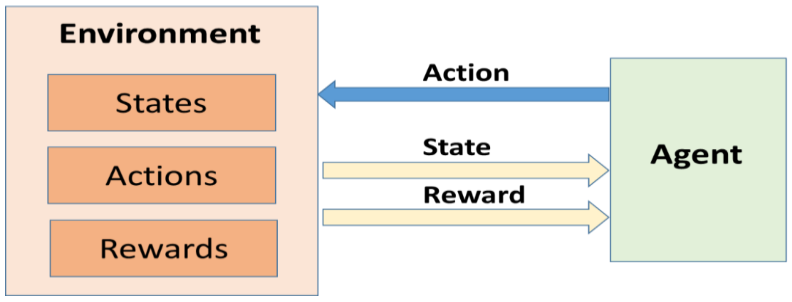
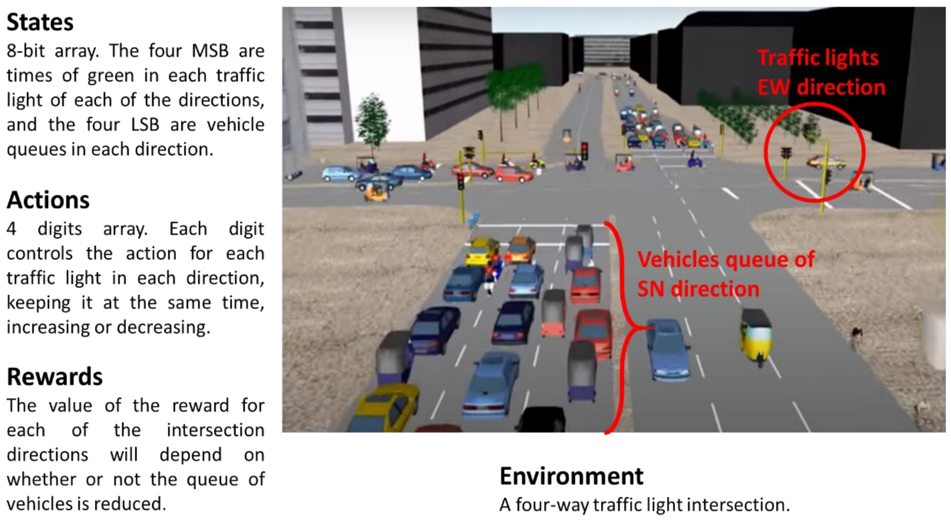


Figure 6. RL process.

The objective of the RL algorithm is to minimize the length of the vehicle queue at intersections, for which the established rewards are the reduction of vehicle queues in each one of the directions. The action taken by the agent generates a new state in the traffic signals (new phase times). This new state will possibly increase or decrease the size of each of the vehicle queues. The algorithm aims to decrease the length of all long queues with the new assigned state, thus, yielding the highest reward. If this does not happen, the reward value obtained will vary (according to the established policies). The algorithm revises the new state of the vehicle queues two signal cycles after the state change.

Figure 7 presents the components of the implemented algorithm for the TSCS. The phases of the traffic signals for a 4-way intersection are effectively at least eight. In some cases, more than eight might be used when additional phases are used to allow special turns. Both the system analysis and the RL traffic signal time recommendation AI algorithm considered eight phases. But four of the eight phases do not change at any time: when the color in any traffic signal is red-yellow, or yellow (red-yellow is used in some countries to indicate that the traffic light will change from red to green). Each one of these static phases was assigned a duration of five seconds, considering the international standard duration for this color in traffic signals is 3–6 s (which is set for an intersection depending on factors such as: design speed, acceleration/decelerate rate, the width of intersection, and reaction time). Table 2 presents the eight considered phases for the chosen 4-way intersection.



States
8-bit array. The four MSB are times of green in each traffic light of each of the directions, and the four LSB are vehicle queues in each direction.

Actions
4 digits array. Each digit controls the action for each traffic light in each direction, keeping it at the same time, increasing or decreasing.

Rewards
The value of the reward for each of the intersection directions will depend on whether or not the queue of vehicles is reduced.

Environment
A four-way traffic light intersection.

Figure 7. RL algorithm elements for TSCS.

Table 2. Phases in 4-way intersection.

Phase Number	Traffic Light 1	Traffic Light 2	Traffic Light 3	Traffic Light 4	Static or Modifiable Time
1	Green	Red	Red	Red	Modifiable
2	Yellow	Red-Yellow	Red	Red	Static
3	Red	Green	Red	Red	Modifiable
4	Red	Yellow	Red-Yellow	Red	Static
5	Red	Red	Green	Red	Modifiable
6	Red	Red	Yellow	Red-Yellow	Static
7	Red	Red	Red	Green	Modifiable
8	Red-Yellow	Red	Red	Yellow	Static

By having four phases with static time, it is only required to define the time of four phases to determine a new time for the eight possible phases. For this reason, it is possible to encode the times of the four phases to which the time is modified as NS, SN, EW, and WE. The code defines the duration of the four phases for which the time can be modified (green light in any of the 4 traffic signals). Time for each direction is defined as either ST or LT (short time or long time). Defining the system in this way simplifies the algorithm and shortens the execution time.

In the developed algorithm, states were identified with an 8-bit array, the four MSB are green light times (0 for short time, 1 for long time) in each traffic signal, for each travel direction (WE, NS, EW, SN); and the four LSB represent vehicle queue lengths for each direction (0 for short queue, 1 for long queue). The actions were identified with a 4-digit array. Each digit controls an action for each traffic signal. If the green time in traffic signal is to be kept, the value of the respective action digit is 0; if the green time is to be decreased the value of the action digit is 1; and if it is to be increased the value is 2. Regarding rewards, if it is possible to decrease the vehicle queue in a certain direction, the most positive value of the reward is taken; if the vehicle queue is maintained, the reward is zero or less than the previous one; finally, if the vehicle queue is increased, the value of the reward is negative. The total reward is calculated by considering the reward obtained in each direction of the intersection.

The implemented algorithm operates in a similar way to the general way presented in Figure 6. The current state of the environment (current plan times and vehicle queue sizes) is the input to the algorithm. The output of the algorithm is the best course of action to perform with the times of each of the traffic signals, indicating whether the time should be maintained, increased, or decreased. The selected action generates a new state in the environment, and this new state is expected to improve traffic conditions.

The developed RL algorithm is based upon Gym from OpenAI [55]. Gym is a toolkit for developing and comparing RL algorithms. The Gym exercise used as a base (called Taxi v3) is related to the movement of a taxicab in a 4×4 matrix which has obstacles in some positions [56]. This exercise was used to facilitate identification and handling of states, actions, rewards, and environment for this specific system, and to identify the way to code this type of algorithm using the Python language.

The implemented algorithm was a Q-learning algorithm. The Q-values were initialized through a training process, which establishes these values depending upon certain basic reward policies. For example, a decrease in a vehicle queue generates a positive reward for the respective direction, while an increase in a vehicle queue generates a negative reward.

This training process takes approximately three to five minutes, depending on the number of training cycles. For the final tests, 10,000 cycles were used, and the training took approximately three minutes. This training would only be performed the first time the algorithm is executed; each time the algorithm is run the Q-values are updated, so the algorithm is constantly learning.

Once training is finished, traffic control operation starts. The agent exposes itself to the environment and receives different rewards by executing different actions. The Q-values are updated using the following equation [57]:

$$Q^*(st, ac) \leftarrow (1-\alpha)Q(st, ac) + \alpha(rw + \gamma \max Q(\text{next state, all actions})) \tag{1}$$

where $Q^*(st, ac)$ is a new Q-value, $Q(st, ac)$ is the current Q-value for a state–action pair, α ($0 < \alpha \leq 1$) is the learning rate, rw is the reward received from the environment, and γ ($0 \leq \gamma \leq 1$) is the discount rate, which determines how much importance the future rewards have. The parameters α (alpha) and γ (gamma) are known as "hyper-parameters" in Q-learning algorithms.

Figure 8 presents a flow chart explaining the algorithm operation. The Q-table is created and updated in an external text file, this allows to load information obtained from previous executions of the algorithm, allowing to continue the learning process. The algorithm is available at [58].

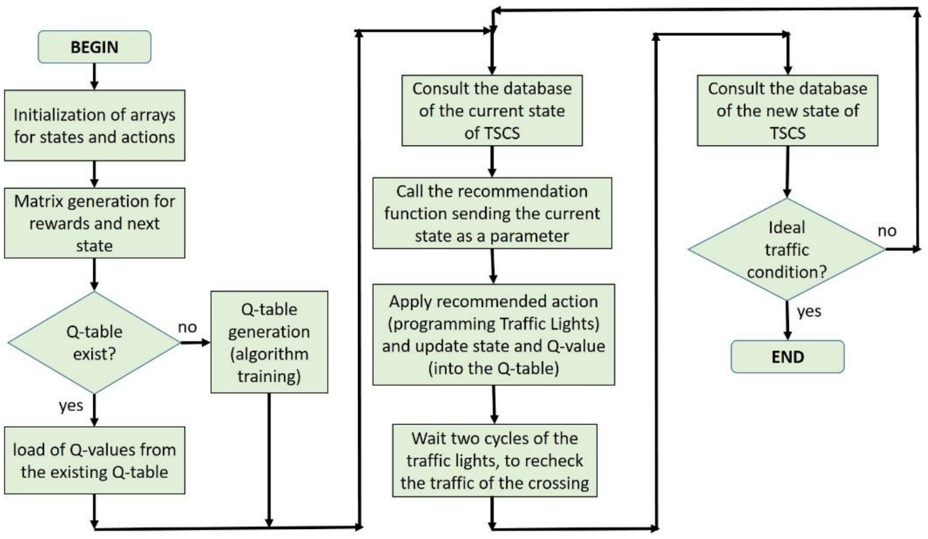

BEGIN
Initialization of arrays for states and actions
Matrix generation for rewards and next state
Q-table exist?
no
Q-table generation (algorithm training)
yes
load of Q-values from the existing Q-table
Consult the database of the current state of TSCS
Call the recommendation function sending the current state as a parameter
Apply recommended action (programming Traffic Lights) and update state and Q-value (into the Q-table)
Wait two cycles of the traffic lights, to recheck the traffic of the crossing
Consult the database of the new state of TSCS
Ideal traffic condition?
no
yes
END

Figure 8. RL algorithm flowchart.

The TSCS system learns during the interaction with the environment in the exploration phase through the algorithm of Figure 8. The algorithm would be executed in a real operating environment (with real traffic flow), but it was not possible to do so, because a traffic signal that could be updated and verified was not available. To verify its correct execution and operation, the values of the new TSCS states were updated in the database using external software. Since it was not possible to test the system in an integrated way with all the modules in a real environment, the development of a simulation through suitable software (SUMO) was proposed; this simulation required adjustments to the initially proposed algorithm which are explained in Section 2.3.3. The simulation made the generation of the Q-table easier; therefore, for a real environment, Q-table training could be performed initially. Such previous training would allow having adequate values before starting the interaction with the environment.

Although the algorithm developed for the real environment could not be tested in an integrated way with the rest of the components developed for the TSCS, it is important to know how it is proposed to learn it, through the generation and permanent updating of the Q- table, which is presented below.

An offline approach to generate the initial Q-table was used, considering a previously recorded response of the underlying traffic process (intersection) to be controlled. This is done to expedite the learning phase required before the system can be implemented.

Subsequently, the Q-table is updated during the interaction with the environment, which can take several minutes, hours, or days of execution in a real traffic environment (depending on the complexity of the environment). For this reason, the developed algorithm gives the option to suspend the process during the exploration phase and resume the update of the Q-table, loading it from a text file (which is continuously updated).

Next, an example of algorithm operation is presented. Bear in mind that the 4 MSB of the state array represent the green light times for each one of the travel directions in the intersection (WE, NS, EW, SN), and the 4 LSB represent the vehicle queue length for each travel direction. Thus, if the 4-way intersection is in the state "10101110", this means that the traffic signals in the WE and EW direction have a long green light time and the traffic signals in the NS and SN directions have a short green light time. In addition, the vehicle queue of the WE, NS, and EW directions exceed the maximum threshold and are considered as long queues, while the vehicle queue in the SN direction is short.

By asking the algorithm for the value of the action that generates the highest reward, the value "0202" could be obtained as an answer. This value causes the green light times of the traffic signals with the WE and EW directions to keep the same value, and the green light time of the traffic signals with the NS and SN directions to increase. With this action, the new status would be "11111110". The algorithm waits for two complete cycles of the traffic signals and rechecks the status at that time to verify if traffic conditions have improved. If conditions improve (e.g., one or more vehicle queues change from long to short) the reward will have a positive value. The Q-table is updated with each update of the status of the intersection.

2.3.3. RL Algorithm Adjustments for Simulation

Once the development of the prototype was finished, unit functional tests of some modules were performed (which are described in Section 2.4). However, testing the operation of the RL traffic signal time recommendation AI algorithm and its interaction with the environment in a real environment was complicated, because special permits were required from government entities in charge of traffic control in a city. In addition, it was important to perform a previous functionality assessment. For this reason, a new version of the algorithm (RL traffic signal time recommendation) was developed to allow simulation through SUMO, a microscopic traffic simulation software [24]. Figure 9 presents a flow diagram explaining the operation of the algorithm allowing the simulation of the TSCS using SUMO. The algorithm can be downloaded at [59]. The version of the algorithm presented in Figure 8 applies to a real operating environment (with real traffic flow), while the version of the algorithm presented in Figure 9 is required for evaluating performance with simulated traffic flows.

The generation of the Q-table in this algorithm (Figure 9) is performed in a similar way to that explained for the algorithm in Figure 8 (Section 2.3.2). The generation of the initial Q-table in this simulation algorithm and its permanent updating during the simulated cycles would allow adequately training the Q-table in a real environment, before starting the interaction.

Nevertheless, the algorithm presented in Figure 9 has important changes compared to the initial version presented in Figure 8. In this version of the algorithm, an integration was made between the initial version (which used only RL) and a SUMO interface, allowing simulation of the online traffic control named TraCI 1.7.0 version (Traffic Control Interface) [60]. TraCI made it possible to measure the vehicle queues (generated in the simulation) and to program the times of the simulated traffic signals in each of the directions (using external files for time plan definition).

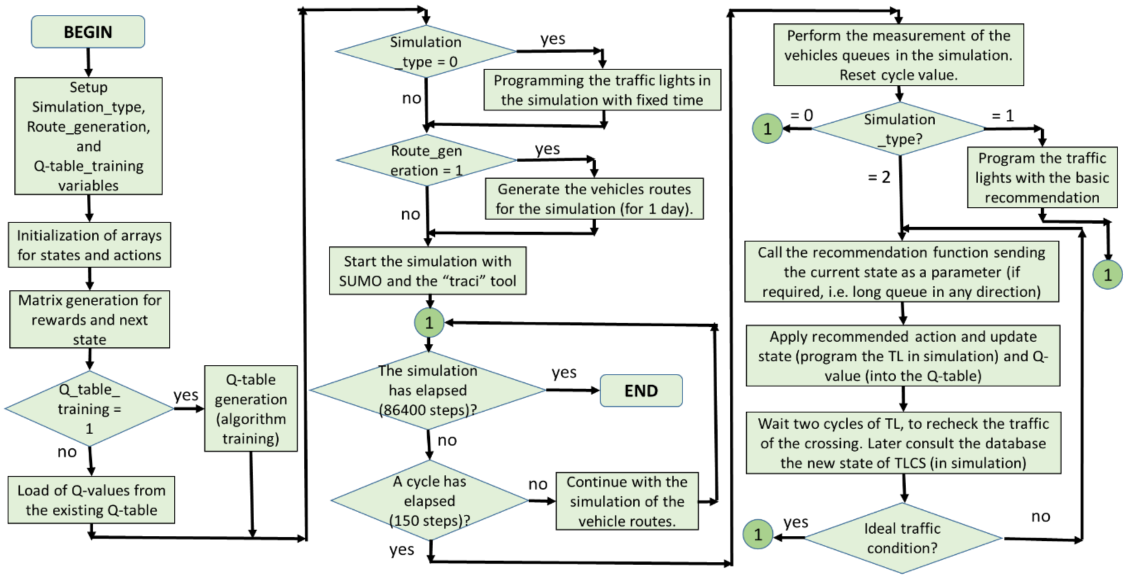

BEGIN
Setup Simulation_type, Route_generation, and Q-table_training variables
Initialization of arrays for states and actions
Matrix generation for rewards and next state
Q_table_ training = 1
yes
no
Q-table generation (algorithm training)
Load of Q-values from the existing Q-table
Simulation _type = 0
yes
no
Programming the traffic lights in the simulation with fixed time
Route_gen eration = 1
yes
no
Generate the vehicles routes for the simulation (for 1 day).
Start the simulation with SUMO and the "traci" tool
1
The simulation has elapsed (86400 steps)?
yes
END
no
A cycle has elapsed (150 steps)?
no
Continue with the simulation of the vehicle routes.
yes
Perform the measurement of the vehicles queues in the simulation. Reset cycle value.
Simulation _type?
= 0
= 1
= 2
1
Program the traffic lights with the basic recommendation
1
Call the recommendation function sending the current state as a parameter (if required, i.e. long queue in any direction)
Apply recommended action and update state (program the TL in simulation) and Q-value (into the Q-table)
Wait two cycles of TL, to recheck the traffic of the crossing. Later consult the database the new state of TLCS (in simulation)
Ideal traffic condition?
yes
1
no

Figure 9. RL algorithm (with SUMO simulation) flowchart.

The traffic flow simulation was compared to two other options to evaluate the performance of the RL algorithm. The first option was the assignment of fixed times in the traffic signals, and the second option was a basic recommendation of traffic signal times, depending on the vehicle queue length for each direction. A setup variable (named "simulation_type") was used to configure the type of simulation to be performed.

The algorithm has two additional setup variables. The second setup variable named "route_generation" determines whether a route file should be generated for the SUMO simulation. The route file determines how many vehicles are going to travel on the simulated roads, along with their speeds and routes. SUMO performs the simulation with the existent route file. If the "route generation" variable is set to 1, a new route file will be generated; if the variable is set to 0, the existing one is used. Finally, the third setup variable named "Q-table_training" determines if a training of the RL algorithm should be performed. If the variable is set to 1, a training of the algorithm is performed, if the variable is set to 0, training is not performed, and the Q-table is loaded from a text file.

The algorithm simulates the steps corresponding to one day (86,400 s); one simulation step in SUMO corresponds to one second. By having different types of simulation, the algorithm allows to simulate traffic behavior at the intersection, using the same vehicle routes and different types of traffic control. The different tests performed with the developed algorithm are explained in Section 2.4.3. The simulated vehicle queue measurement is performed every 150 simulation steps, which corresponds to approximately two cycles of the traffic signals.

2.4. *Designing and Developing of System Tests*

The tests performed on the TSCS were:

- Design and development of computer vision tests (for calculating vehicle queues);
- Design of captured video signals and the planned data set;
- Design and development of TSCS simulation tests using SUMO;
- Web application testing;
- Real-time module integration testing.

Each of them is described in detail below.

2.4.1. Design and Development of Computer Vision Tests (for Calculating Vehicle Queues)

In the TSCS prototype, the measurement of vehicle queues was proposed through the ITS RE RTM module. This module includes a vehicle queue AI algorithm which receives the video signals of the traffic of the four directions of an intersection as input and performs the periodic measurement of the vehicle queue in each direction. The algorithm was developed and tested in stages, increasing the complexity of the functionality and scaling the number of signals evaluated in each stage. Initially, the algorithm was tested using some publicly available images of roads, each with a considerable number of vehicles, analyzing the images one by one. Then, publicly available video files of traffic cameras at different intersections were used, measuring the vehicle queues periodically. These measurements were analyzed, reviewing which model offered by YOLO could have the best ratio between processing time and fidelity of detection results. These tests were also done by reviewing the videos one by one. Later, the counting was tested with a video taken and streamed from a USB camera connected to the Raspberry Pi SBC (in a line of vehicles not located at a traffic signal intersection). Finally, this process was scaled up to stream two videos simultaneously.

To test the vehicle queue algorithm in a 4-way intersection with traffic signals, an intersection in the city of Popayán was chosen as a case of study. This point has the needed characteristics and presents a high level of traffic. The selected point is located at latitude 2.459476 N and longitude 76.597016 W. Video was captured in the four directions of the intersection, and these videos were used for the final tests of the vehicle counting algorithm. The detailed information about design of captured video signals and the planned structure of the data set is presented in Section 2.4.2.

Two commonly used metrics in this type of research, precision and recall [61,62], were used to evaluate the performance of the computer vision algorithm for vehicle detection. Before presenting the definition of these two metrics, it is important to clarify the following definitions regarding the results obtained at the time of detecting a vehicle in an image or video using the algorithm [61,62]:

- True positive (TP): the algorithm detects a vehicle, and a vehicle is present.
- False positive (FP): the algorithm detects a vehicle, and a vehicle is not present.
- False negative (FN): the algorithm does not detect a vehicle, and a vehicle is present.
- True negative (TN): the algorithm does not detect a vehicle, and a vehicle is not present.

Precision is the ratio of the number of TP to the total number of positive predictions. The formula is as follows.

$$\text{Precision} = (\text{TP})/(\text{TP} + \text{FP}) \tag{2}$$

Recall is the ratio of the number of true positives to the total number of actual (relevant) objects:

$$\text{Recall} = (\text{TP})/(\text{TP} + \text{FN}) \tag{3}$$

These two metrics were calculated for the final three types of computer vision tests, and the results obtained in each of them are presented in Section 3.1.

2.4.2. Design of Captured Video Signals and the Planned Data Set

To take the videos in the selected intersection, a series of recording sessions were performed, which consisted of taking videos of three cycles per traffic signal. Considering that the intersection has four directions (four traffic signals), each session required taking four videos, each one with a duration of three cycles, which was estimated at approximately 2 to 5 min. The duration of a session was estimated at approximately one hour. The plan for capturing the videos was to film on each of the seven days of the week (Monday to Sunday) and to hold a session in the morning and another in the afternoon, trying to do the filming at the times with peak traffic flow. Therefore, a total of approximately 14 h was planned for the 14 sessions. The principal limitations considered were unfavorable weather, achieving a good location (a high vantage point) to record the traffic, and visualizing the entire queue in each direction of the intersection. Videos were ordered by the day of the week, then by the time range of the day (morning or afternoon), and then by the direction at the intersection. Two months were allocated to perform the entire process, considering that filming would not take place on consecutive days.

2.4.3. Design and Development of TSCS Simulation Tests Using SUMO

The RL algorithm was tested and modified several times to best approach a real operating environment. The hyper-parameters alpha and gamma were tuned reviewing the values obtained in the simulation (in terms of vehicle queue length) in the simulation of one of the days. The behavior of the system was changed by increasing or decreasing the hyper-parameters (which can take values between 0 and 1), seeking to obtain the lowest mean value for vehicle queue length. Best results were obtained with alpha = 0.6 and gamma = 0.4. With alpha greater than 0.6, the Q-values were updated too frequently, which generated a great variety of recommended states for the same initial state, while with values lower than 0.6 the recommended states did not vary enough to obtain a significant improvement in the results. Moreover, as more repetitions of the simulation were executed (updating the Q-table each time), alpha could be assigned a lower value, because the knowledge base was growing. Regarding the gamma values, for values higher than 0.4, the algorithm tended to give too much importance to rewards of future states, which is not favorable because the objective is to find the best reward for the next state. While for values lower than 0.4 the reward was not significant, because an immediate reward in this case is not possible. Considering that the reward would be evaluated two cycles after the state change in the traffic signals, the ideal value was an intermediate value (0.4).

The chosen real intersection was represented in SUMO, considering the dimensions of the roads, the number of lanes in each direction, and the current phases of the traffic signals. Figure 10 presents the four directions of the intersection represented in SUMO, along with the vehicular flow, which accounts for differently sized vehicles (cars are yellow, buses are red, and trucks are green). The light color of each traffic signal is represented as a line across the road, before entering the intersection. Some aspects of the intersection such as U-turns were not considered. Figure 10 indicates the maneuvers a vehicle can perform at the intersection: go straight ahead, turn left, or turn right.

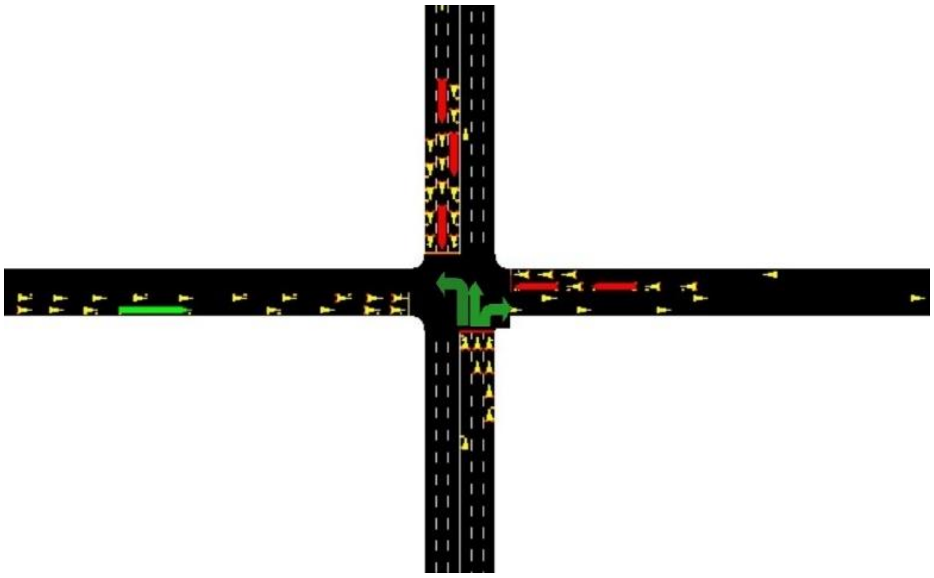


Figure 10. Screenshot of RL algorithm simulation (using SUMO) indicating the maneuvers a vehicle can perform at the intersection. The yellow triangles on the roads simulate small vehicles, the red rectangles simulate buses, while the green rectangles simulate large trucks. Green arrows indicate the maneuvers a vehicle can perform at the intersection.

The information collected in the video signal data set and the vehicle queue algorithm was used to simulate the vehicle frequency in each direction of the intersection. Operation was simulated for 5 days (86,400 execution steps in SUMO were equivalent to 86,400 simulated seconds, which is equivalent to the time of one day). Simulation was performed as follows:

- Initially, the setup variables were set to the following values: simulation_type was set to 2, route_generation to 1, and Q-table_training to 1. This way, the simulation of the first day of TSCS operation was performed using the RL recommendation, generating a new vehicle routing file and training the algorithm's Q-table.
- Once the simulation of the first day was done, results were compared to the other two types of simulations (basic recommendation and fixed times) for the first day. To perform the other two simulations, the setup variables were configured as follows: for the second simulation (basic recommendation) simulation_type was set to 1, route_generation to 0, and Q_table_training to 0; for the third simulation (fixed times) simulation_type was set to 0, and the other two were unchanged. This guaranteed that results obtained with the three different types of simulation were tested, for the same vehicle routes in a day.
- To simulate days 2, 3, 4, and 5, the first two steps were repeated. The algorithm guarantees randomness in the generation of vehicle routes, so as not to repeat the same route file on the other days. Each file generated from the routes has a random frequency of vehicles in each direction, at each time range of the day (7 time ranges per day were used: 0–6 a.m., 6–9 a.m., 9–11 a.m., 11 a.m.–2 p.m., 2–5 p.m., 5–8 p.m., 8 p.m.–0 a.m.). In this way, an attempt was made to approximate the behavior of a real environment, in which the frequency of vehicles is similar from day to day of the week, but with certain changes between hour ranges.

Once all the algorithm simulations had been performed, the information was analyzed and graphed. These results can be reviewed in Section 3.3.

2.4.4. Web Application Testing

The web application tests were designed according to the functionality identified with the prototype development methodology (Scrum). Through Scrum, the "user stories" of each one of the actors were identified. This same functionality determined by the "user stories" was verified through the tests of the web application. Each "user story" has one or more associated interfaces of the web application, therefore, when the functionality of the "user story" is verified, the interfaces of the web application were verified to operate as expected. Some additional "user stories" were related to other functional modules. For example, the "user stories" related to maps required the use of Google APIs, also the "user stories" related to the algorithms required verifying the interaction between the web application and such algorithms.

2.4.5. Real-Time Module Integration Testing

Finally, once the TSCS simulation yielded satisfactory results, some real-time tests of the system were performed to verify that the operating times were adequate and the system was able to operate in a production environment. The ITS RE RTM module was tested by inputting four video signals simultaneously, these four video signals were used to estimate the vehicle queue lengths by the AI algorithm applied to count the vehicles in each of the directions of the intersection and report the information to the database. The TSCS web interface was used to query the status of each of the traffic signals and the latest vehicle queue data in each direction of the intersection. Then, through the web interface, the execution of the recommendation algorithm was requested, and execution time was measured. This test was performed 20 times measuring the response time for vehicle queue length calculation and the response time of the recommendation algorithm. It was not possible to test the update of the traffic signal times, because it was not possible to obtain the necessary test infrastructure, nor was it possible to obtain authorization from the city government to test the system in a real setting.

3. Results

The results obtained in each of the groups of tests performed are presented below.

3.1. Computer Vision Results (Measurement of Vehicle Queues)

This Subsection presents the results of the tests outlined in Section 2.3.1. The first stage of testing, where publicly available images were used, allowed confirming that YOLOv3 adapts to the needs required by the TSCS for recognition and counting of vehicles, as it is an extremely fast and accurate model. At this stage, the metrics mentioned in Section 2.3.1 were not used.

In the second stage, with the help of publicly available traffic videos, 3 videos with an average duration of 60 s were reviewed, considering a periodicity of 10 s for measurement of the vehicle queue. From this stage onwards, the precision and recall metrics were used to evaluate the performance of the algorithm. At this stage, a comparison was made between two models (YOLOv3-320 and YOLOv3-tiny) to define the cost of machine vs. effectiveness of each model. Results showed that the best model to work with a Raspberry PI SBC was the YOLOv3-320 (which counts vehicles at 45 frames per second, and has a much lower error rate) with an average precision of 100% and an average recall of 90.28%. The results obtained with the YOLOv3-tiny model were very low in terms of recall, with an average of 29.72%. The results obtained for vehicle counting with the YOLOv3-320 model are presented in Table 3.

Table 3. Test results for publicly available traffic videos.

Video Number	Number of Analyzed Image	Real Number of Vehicles	Number of Detected Vehicles	Precision	Recall
1	1	8	7	100%	87.5%
1	2	18	16	100%	88.88%
1	3	6	6	100%	100%
2	1	31	26	100%	83.87%
2	2	35	29	100%	82.85%
3	1	4	4	100%	100%
3	2	9	8	100%	88.88%

For the third stage of testing, five videos taken at the selected intersection was used. The total duration of the videos was 700 s approximately, and a vehicle queue count was performed every 20 s. Results showed an average precision of 100% and an average recall of 94.34%. The results of this third stage are presented in Table 4.

Table 4. Video test results for selected intersection traffic.

Video Number	Number of Analyzed Image	Real Number of Vehicles	Number of Detected Vehicles	Precision	Recall
1	1	4	4	100%	100%
1	2	5	5	100%	100%
1	3	11	10	100%	90.90%
1	4	9	8	100%	88.88%
1	5	8	8	100%	100%
1	6	11	10	100%	90.90%
2	1	12	11	100%	91.66%
2	2	11	10	100%	90.90%
2	3	4	4	100%	100%
2	4	10	9	100%	90%
2	5	13	12	100%	92.30%
2	6	8	7	100%	87.5%

Table 4. *Cont.*

Video Number	Number of Analyzed Image	Real Number of Vehicles	Number of Detected Vehicles	Precision	Recall
3	1	7	7	100%	100%
3	2	12	11	100%	91.66%
3	3	10	9	100%	90%
3	4	14	14	100%	100%
3	5	6	6	100%	100%
3	6	12	11	100%	91.66%
3	7	8	8	100%	100%
4	1	10	9	100%	90%
4	2	5	5	100%	100%
4	3	16	15	100%	93.75%
4	4	10	9	100%	90%
4	5	10	10	100%	100%
4	6	9	8	100%	88.88%
4	7	7	7	100%	100%
5	1	15	13	100%	86.66%
5	2	12	12	100%	100%
5	3	16	15	100%	93.75%
5	4	9	8	100%	88.88%
5	5	8	8	100%	100%
5	6	11	10	100%	90.90%

The results presented in Tables 3 and 4 correspond to tests performed to evaluate correct operation of the algorithm. Table 3 presents results using videos from the Internet, and Table 4 uses videos feeds from the selected intersection. These tests used a low number of images (one image every 20 s of video) and were useful to adjust some algorithm parameters and to perform the final tests (fourth stage). Stage 4 analyzed 50 images generated by reviewing a considerable number of videos (more than 1000 s of videos were reviewed to obtain the 50 collected images) directly at the intersection in the four directions (also taking an image every 20 s of each video).

At the stage 4 testing, a simultaneous counting of two videos was performed to test the scalability of transmission and counting of vehicle queues for the selected intersection, in this case using the data set of collected traffic videos. A comparison of the number of vehicles present at the intersection with respect to the vehicles counted by the algorithm in 50 images was made. Results of stage 4 are presented as a bar chart (Figure 11), where each bar represents the value for the recall metric. The precision metric of the algorithm for each of the 50 measurements was 100%, because in no case was a vehicle identified in the image, in a place that did not correspond.

An average recall of 92.67% was obtained in the final tests performed on the counting algorithm, which is considered acceptable for the TSCS prototype operation. In total, 14 out of the 50 analyzed images yielded a correct vehicle count, whereas in 36 of the 50 images the number of real vehicles was greater than the vehicles counted by the algorithm, with an average of two non-detected vehicles. In Figure 12, a screenshot at the time of performing one of the final stage tests, where the videos captured in the selected intersection were used, is shown.

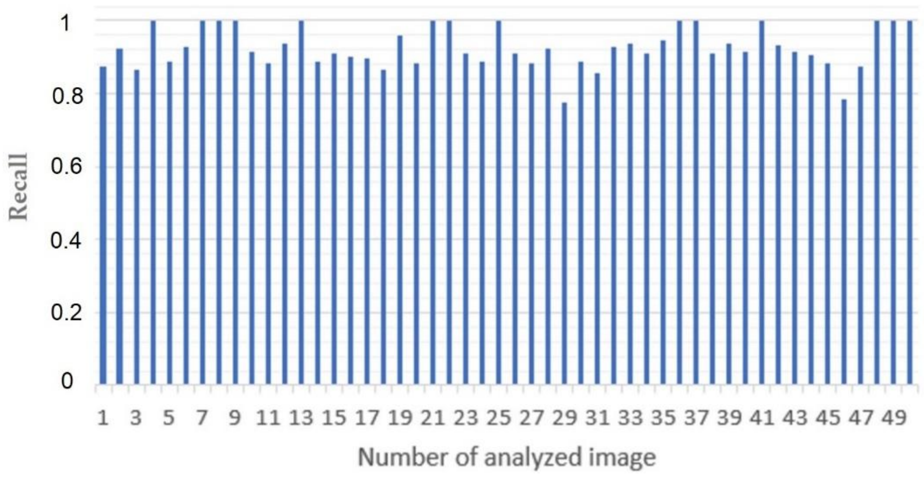
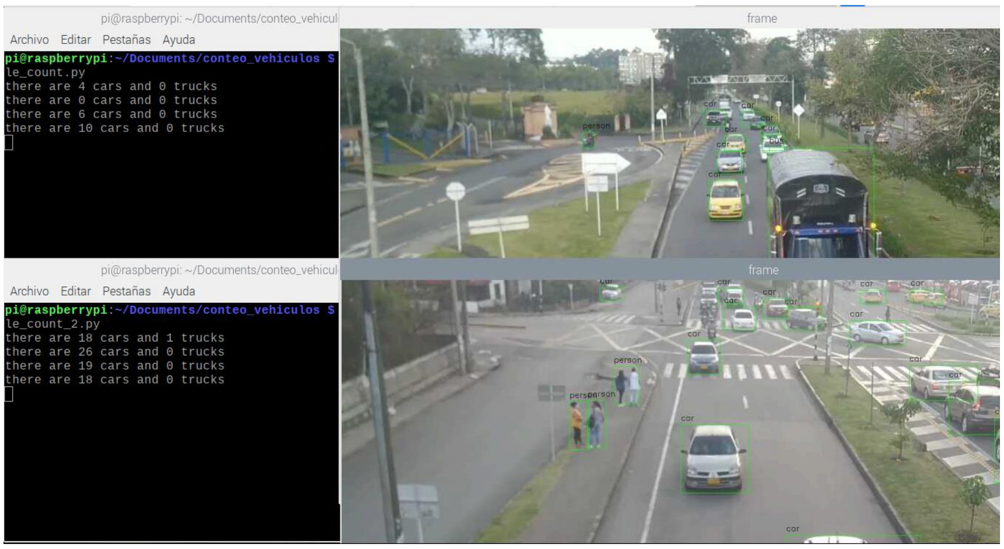


Figure 11. Results of the counting algorithm used on data set (stage 4 of tests of the measurement vehicles queue).


pi@raspberrypi: ~/Documents/conteo_vehicul
Archivo Editar Pestañas Ayuda
frame
person
car
pi@raspberrypi: ~/Documents/conteo_vehicul
Archivo Editar Pestañas Ayuda
frame

```
pi@raspberrypi:~/Documents/conteo_vehiculos $
le_count.py
there are 4 cars and 0 trucks
there are 0 cars and 0 trucks
there are 6 cars and 0 trucks
there are 10 cars and 0 trucks
```

```
pi@raspberrypi:~/Documents/conteo_vehiculos $
le_count_2.py
there are 18 cars and 1 trucks
there are 26 cars and 0 trucks
there are 19 cars and 0 trucks
there are 18 cars and 0 trucks
```

Figure 12. Screenshot: fourth stage of tests, measurement of vehicle queue.

3.2. Data Set of Collected Video Signals

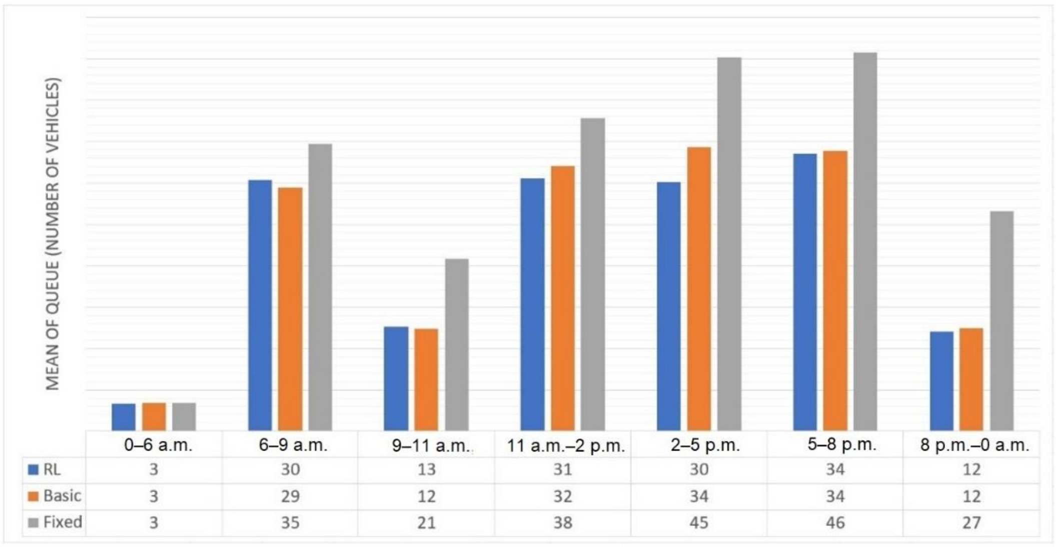
This Subsection presents the results of video data set planned in Section 2.3.2. The data set of video signals was organized as follows. The first classification criterion was the day of the week; videos were not taken in consecutive weeks to increase data randomness. Within each day the videos were taken in two sessions, morning and afternoon (second criterion). The files were named with the following structure: travel direction in the intersection, (e.g., N-S which means that the shot was taken from cars coming from north to south), followed by the acronym of the day and finally the time at which videos were taken. Each video contains three cycles of green lights for each direction, yielding an approximate duration of 3:30 for the N-S and S-N directions and 1:40 for the E-W and W-E directions. A total of 56 videos were collected. Videos were coded in H.264 format at 480p resolution, because they allow an accurate count with the YOLO tool and do not consume a large amount of storage. This data set can be accessed at the following Google drive link: https://bit.ly/3apf7i6 (accessed on 10 August 2021).

These videos were used to test the measurement algorithm. This data set can be used for future mobility projects related to traffic measurement or vehicle detection.

3.3. Results of TSCS Simulation Test Using SUMO

This Subsection presents the results of the tests outlined in Section 2.3.3. Three types of simulations were performed through using SUMO and the algorithm explained in Figure 9. To change the type of simulation, the parameter named "simulation_type" was used. The "RL" simulation (setting simulation_type to 2) is used by the recommendation function to obtain the new state in the traffic signals (with the Q-table). The "basic" simulation (setting simulation_type to 1) changes the state of a traffic light (in a certain direction) to a long time (in green) if the queue of vehicles is long, and vice versa. Finally, the "fixed" simulation (setting simulation_type to 0) uses fixed green light times for each of the intersection traffic signals. The fixed times for each direction of the intersection were defined by simulation operation tests to allow the best operation. The ideal value for this parameter was 20 s in all directions.

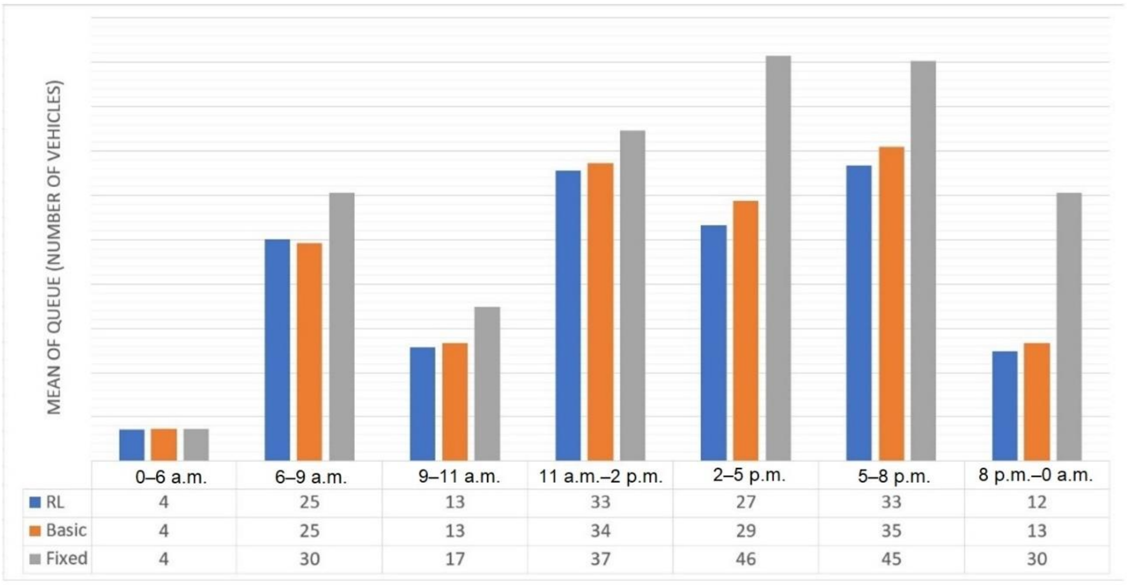
Initially, the objective of reducing vehicle queues in all directions was tested. The algorithm used the vehicle queues measured every two cycles of the traffic signals in the TSCS database. Measurements were made for each of the three types of simulation ("RL", "basic", and "fixed") during five days of the week. Some results obtained with the vehicle queue measurements (day 1 and day 2) for the three simulation types are presented in Figures 13 and 14. The results show that the mean size of vehicle queues obtained in the "RL" simulation type was shorter than the mean size of vehicle queues obtained in the "fixed" simulation (28.83% shorter for day 1 and 29.66% shorter for day 2). In addition, the mean size of vehicle queues for the "RL" simulation was shorter (2% for day 1 and 4% for day 2) than the mean size of vehicle queues obtained in the "basic" simulation.



	0–6 a.m.	6–9 a.m.	9–11 a.m.	11 a.m.–2 p.m.	2–5 p.m.	5–8 p.m.	8 p.m.–0 a.m.
■ RL	3	30	13	31	30	34	12
■ Basic	3	29	12	32	34	34	12
■ Fixed	3	35	21	38	45	46	27

Figure 13. Mean size of vehicle queues, on day 1 by time ranges, for each type of simulation.

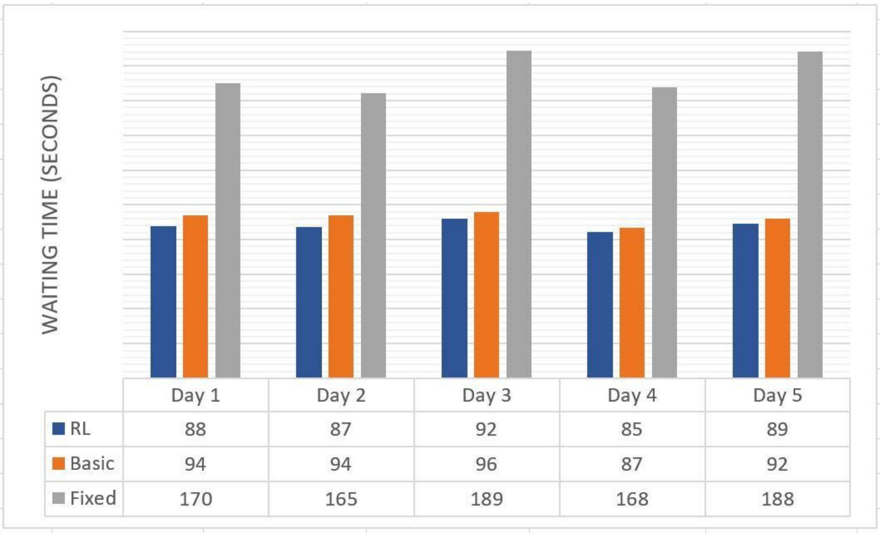
Although the difference between the vehicle queue size of each of the simulation types verifies the fulfillment of the objective of the RL algorithm, it does not represent information that can be analyzed in detail. Therefore, other variables were searched in the simulation results file (generated by SUMO for each simulation run). The variables selected from the file were waiting time and time loss. A low value of these variables means better traffic behavior at the intersection, because vehicles must wait less time at the intersection and lose less time moving. These variables are directly related to the vehicle queues, because if the vehicle queue becomes shorter, the waiting time and time loss decrease accordingly.



	0–6 a.m.	6–9 a.m.	9–11 a.m.	11 a.m.–2 p.m.	2–5 p.m.	5–8 p.m.	8 p.m.–0 a.m.
🟦 RL	4	25	13	33	27	33	12
🟧 Basic	4	25	13	34	29	35	13
⬜ Fixed	4	30	17	37	46	45	30

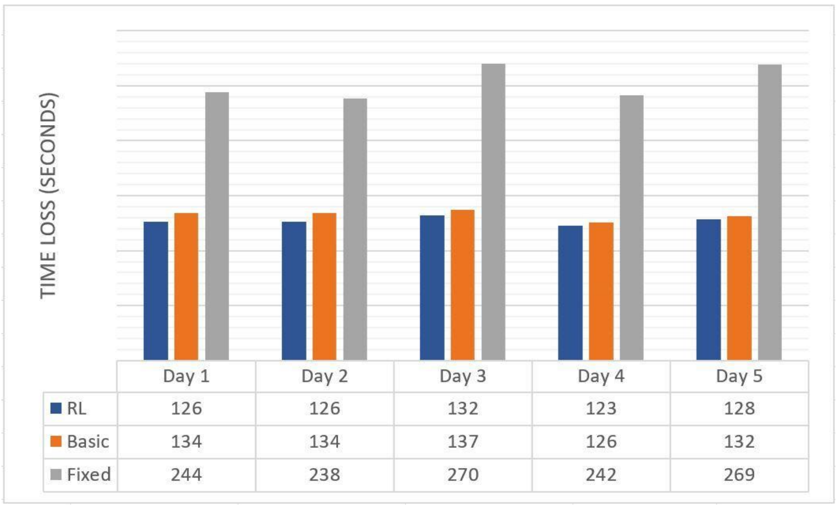
Figure 14. Mean size of vehicle queues, on day 2 by time ranges, for each type of simulation.

The results file has a value of these variables for each vehicle (in seconds). To compare the three types of simulation in traffic control, the mean of the values of the variables was calculated for each day, both for each of the time ranges, and for the entire day. Figures 15–20 present some of the obtained results. Figures 15 and 16 present the mean of waiting time and loss time (in seconds) of the vehicles, during 5 days, for each type of simulation. Figures 17 and 18 present the mean of waiting time (in seconds) for each of the time ranges, for two days. Figures 19 and 20 present the mean of time loss (in seconds) for each of the time ranges, for the same two days. Figures 15 and 16 show that simulation through the RL recommendation type has a better behavior (lower values) in terms of waiting time, when compared to the other two types of simulation.



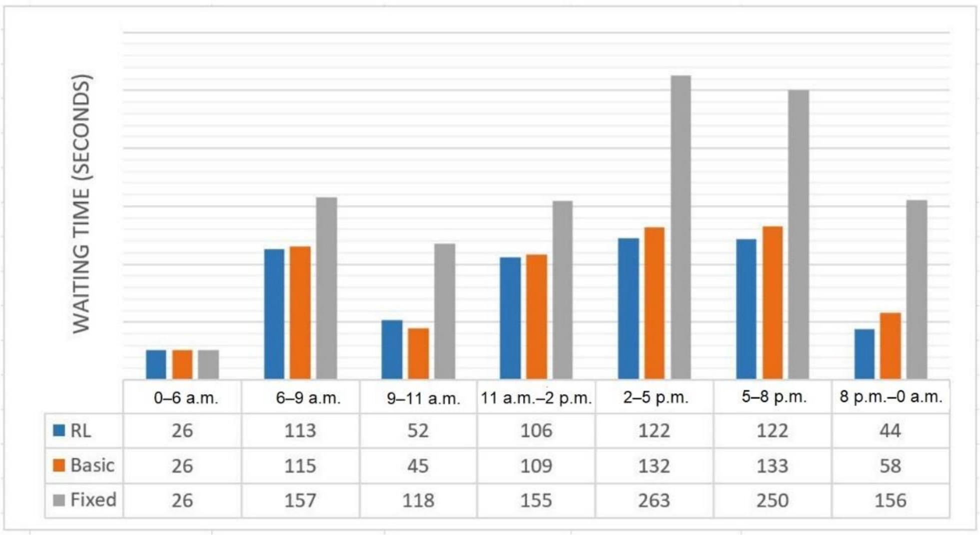
	Day 1	Day 2	Day 3	Day 4	Day 5
🟦 RL	88	87	92	85	89
🟧 Basic	94	94	96	87	92
⬜ Fixed	170	165	189	168	188

Figure 15. Mean of waiting time (in seconds) of the vehicles, during 5 days, for each simulation.



	Day 1	Day 2	Day 3	Day 4	Day 5
🟦 RL	126	126	132	123	128
🟧 Basic	134	134	137	126	132
⬜ Fixed	244	238	270	242	269

Figure 16. Mean of time loss (in seconds) of the vehicles, during 5 days, for each simulation.



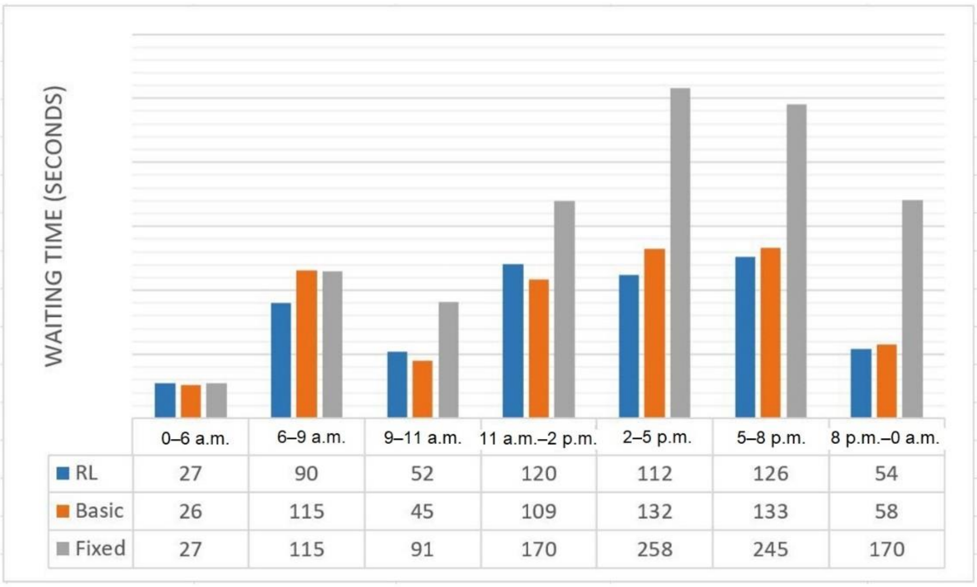
	0–6 a.m.	6–9 a.m.	9–11 a.m.	11 a.m.–2 p.m.	2–5 p.m.	5–8 p.m.	8 p.m.–0 a.m.
🟦 RL	26	113	52	106	122	122	44
🟧 Basic	26	115	45	109	132	133	58
⬜ Fixed	26	157	118	155	263	250	156

Figure 17. Mean of waiting time (in seconds) of the vehicles, on day 1 by time ranges, for each simulation.

Figures 17 and 18 show the behavior of the three types of simulations by time range, for the waiting time variable. Figure 17 shows the results of day 1 and Figure 18 the results of day 2. The charts show better values for most of the time ranges when the RL recommendation type simulation is used.

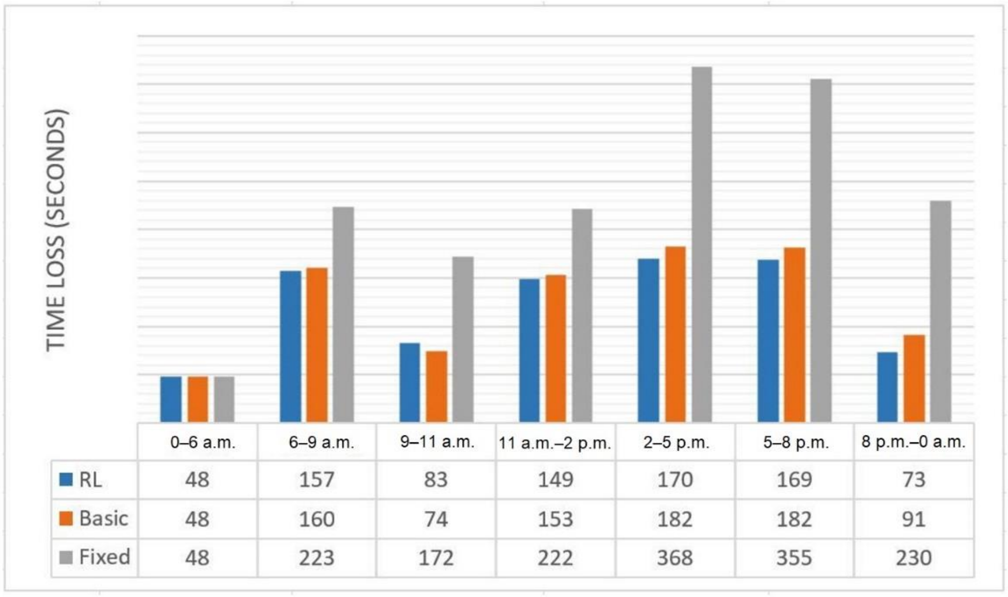
Figures 19 and 20 show the behavior of the three types of simulations by time range, for the time loss variable. Behavior is like that of the waiting time variable.

The RL recommendation for changing the times of the phases of the traffic signals is better when compared to the traffic control with fixed times, in percentages ranging from 46.86% to 52.70% (49.79% on average for the five days), which is a considerable improvement. The RL recommendation is also better with respect to traffic control with the basic recommendation, in percentages ranging from 2.69% to 6.74% (4.6% on average for the five days).



	0–6 a.m.	6–9 a.m.	9–11 a.m.	11 a.m.–2 p.m.	2–5 p.m.	5–8 p.m.	8 p.m.–0 a.m.
🟦 RL	27	90	52	120	112	126	54
🟧 Basic	26	115	45	109	132	133	58
⬜ Fixed	27	115	91	170	258	245	170

Figure 18. Mean of waiting time (in seconds), on day 2 by time ranges, for each type of simulation.

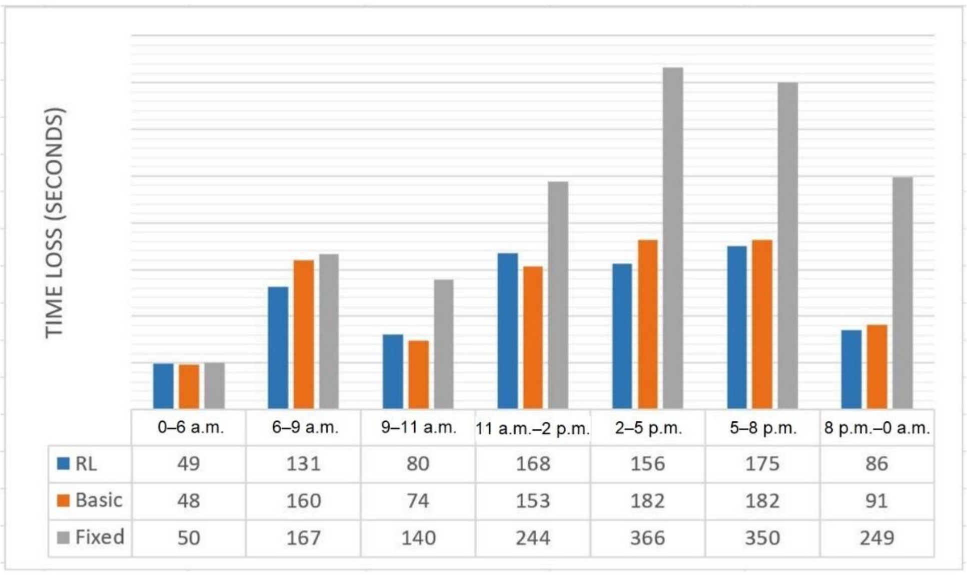


	0–6 a.m.	6–9 a.m.	9–11 a.m.	11 a.m.–2 p.m.	2–5 p.m.	5–8 p.m.	8 p.m.–0 a.m.
🟦 RL	48	157	83	149	170	169	73
🟧 Basic	48	160	74	153	182	182	91
⬜ Fixed	48	223	172	222	368	355	230

Figure 19. Mean of time loss (in seconds), on day 1 by time ranges, for each type of simulation.

Regarding time loss, the algorithm using the RL recommendation simulation type is also better compared to the traffic control with fixed times in the traffic signals, in percentages ranging from 46.98% to 52.32% (49.65% on average for the five days). The RL recommendation is also better with respect to traffic control with the basic recommendation, in percentages ranging from 2.45% to 5.75% (4.1% on average for the five days).

The behavior of the simulations performed during each of the five days is similar, approximately the same percentages of difference between the three types of simulation are maintained on each day. This behavior occurs in the two variables (waiting time and time loss). This indicates that the behavior of the RL recommendation for the times of the traffic signals operates properly even if there are changes in the routes and frequencies of the vehicles.



	0–6 a.m.	6–9 a.m.	9–11 a.m.	11 a.m.–2 p.m.	2–5 p.m.	5–8 p.m.	8 p.m.–0 a.m.
🟦 RL	49	131	80	168	156	175	86
🟧 Basic	48	160	74	153	182	182	91
⬜ Fixed	50	167	140	244	366	350	249

Figure 20. Mean of time loss (in seconds), on day 2 by time ranges, for each type of simulation.

3.4. Results of the Web Application Operation Tests

This Subsection presents the results of the tests outlined in Section 2.4.4. The functionality planned for each of the users of the web application was successfully developed. Integration of the web application with the two developed algorithms (vehicle queue and RL traffic signal recommendation) required some adjustments, because some minor inconveniences were detected during the initial tests. All other operational tests of the web application functionality were successful. Figures 21–23 show screenshots of the web application interfaces for some of the main TSCS functions. Figures 21 and 22 correspond to administrator-type user functionality, whereas Figure 23 corresponds to functionality of the TOP type user. Figure 21 shows a map of the city. The red markers pinpoint the location of the intersections managed by the system.

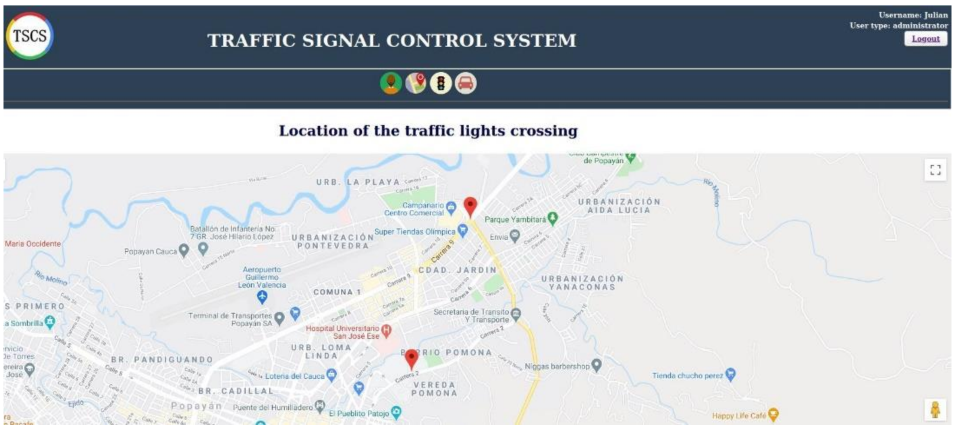
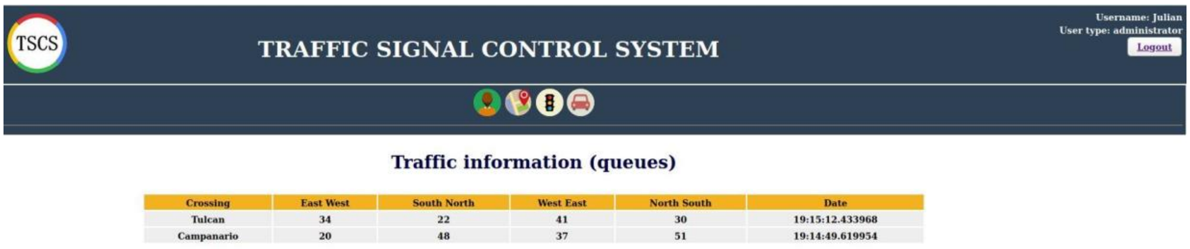


Figure 21. Screenshot: Location of managed intersections.

Figure 22 shows a table of vehicle queue sizes at the managed intersections. This information is updated in real time.

Figure 23 shows the interface for the RL algorithm execution at a given intersection. The interface shows the current green light times and the queue size for each direction. The new green light time recommendations are also shown, and the TOP user is asked to confirm the update.



TSCS

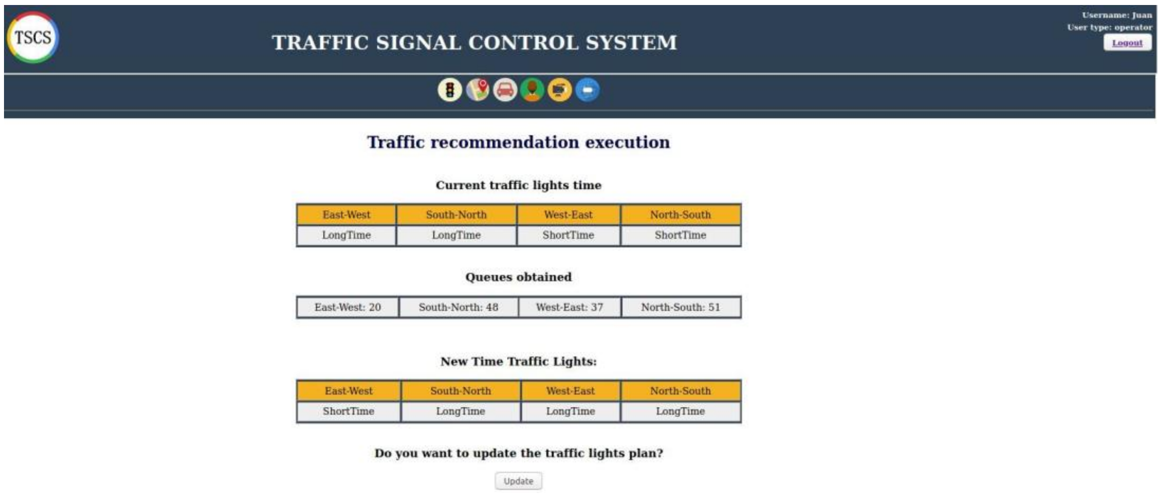
TRAFFIC SIGNAL CONTROL SYSTEM

Username: Julian
User type: administrator
Logout

Traffic information (queues)

Crossing	East West	South North	West East	North South	Date
Tulcan	34	22	41	30	19:15:12.433968
Campanario	20	48	37	51	19:14:49.619954

Figure 22. Screenshot: Traffic information.



TSCS

TRAFFIC SIGNAL CONTROL SYSTEM

Username: Juan
User type: operator
Logout

Traffic recommendation execution

Current traffic lights time

East-West	South-North	West-East	North-South
LongTime	LongTime	ShortTime	ShortTime

Queues obtained

East-West: 20	South-North: 48	West-East: 37	North-South: 51

New Time Traffic Lights:

East-West	South-North	West-East	North-South
ShortTime	LongTime	LongTime	LongTime

Do you want to update the traffic lights plan?

Update

Figure 23. Screenshot: Time recommendation execution.

3.5. Results of Real-Time Module Integration Testing

The real-time module integration testing aimed to evaluate the way in which the different components interact and exchange information. Additionally, they were performed to determine the response times of the TSCS in an operating environment at an intersection. This test was performed 20 times measuring the response time of the vehicle queue measurement algorithm using four video signals (one signal in each direction of the intersection) and the response time of the recommendation algorithm.

The 4 vehicle queues were calculated in an average of 1.57 s. Each video feed captures an image approximately every 20 s to be analyzed by the algorithm. The selection of the image in each of the four video feeds can be done in parallel to save time. Data is recorded in the database when the last queue is processed; the maximum delay time between the start of the measurement and the data recorded in the database was 2.59 s on average, for the 20 tests performed. The mean response time of the time recommendation algorithm was around 7 s. These time delays are adequate because they are much lower than the cycle times of the traffic signals at the intersection (150 s) and than the time for measuring the vehicle queues (20 s).

4. Discussion

The ITS architecture designed for the TSCS is based upon international reference architectures (ARC-IT and FRAME) and the context of a city in a developing country. This ITS architecture of the TSCS can be used by projects in cities that have a similar context, for development of prototypes of this system. Likewise, the procedure used for the design of the ITS architecture can be taken as a model for design of an architecture of another system related to ITSs in future research.

The developed TSCS prototype used the proposed ITS architecture as its main source. In addition, specific technologies and hardware elements were proposed, considering budget limitations for a city in a developing country. The prototype was developed and tested with good results; therefore, it can be taken as a reference for developments in similar cities. Although the traffic control tests of the prototype were simulated due to logistical limitations of performing such control tests in a real environment, an attempt was made to reflect an environment as close to reality as possible.

Analyzing the results obtained from the simulation, it was possible to show that the system had similar results to works reviewed in literature related to RL (such as [4], [5]). The developed prototype improves traffic (specifically, waiting time and time loss of vehicles) at a traffic signal intersection by approximately 50% when compared to traffic control performed with traffic signals using fixed green light times. This percentage is similar to those obtained in related research. However, the prototype in this work uses a simpler, standardized architecture, featuring viable components and low complexity algorithms for determination of state-actions, coding and execution efficiency.

The results of the proposed prototype were also compared to the use of a basic recommendation algorithm. The waiting and loss times for the traffic control prototype were approximately 5% lower than those obtained with the basic recommendation, which represents a smaller scale, nevertheless significant improvement.

The improvement reported in other works related to traffic control using RL shows that using complex proposals (in terms of determination of states, actions, and rewards and more robust algorithms) yields not so significant improvements, with a low percentage of reduction of times of vehicles at the intersection. Therefore, it is considered that the percentage of improvement of this prototype is significant, considering its low complexity and limited implementation budget.

The developed prototype implemented two AI algorithms with very good results. The vehicle queue algorithm allowed counting the waiting vehicles in each direction of the intersection, with an average recall of 92.67% and average precision of 100%. The values of these metrics are considered as sufficient and adequate.

The RL recommendation time algorithm was developed in two versions. The first version allowed integration with the prototype in production, but it was not possible to test it in a real environment (due to the already mentioned logistical limitations). The second version allowed simulation of the prototype using the SUMO software, yielding positive results.

The hyper-parameter values of the RL recommendation time algorithm were modified, seeking the best results in the simulation in terms of vehicle queue length. The values of the hyper-parameters yielding the best results were alpha = 0.6 and gamma = 0.4.

Once the algorithms were independently tested, real-time integration tests were performed to determine if the response time was adequate. The results obtained showed that the maximum delay time between the start of the measurement of the four vehicle queues and the values being recorded in the database was 2.59 s on average, for the 20 tests performed. The response time mean of the time recommendation algorithm was 7 s. These times were considered adequate, because the time of a traffic signal cycle is approximately 150 s and the period between queue measurement is approximately 20 s. The system operator (TOP) can request for the execution of the time recommendation algorithm at any time, which has as input information from the vehicle queue (data collected 23 s ago or less), and the response is obtained in approximately 7 s.

Another significant result of the developed work was the compiled data set of video signals. These videos allow visualizing the behavior of current traffic in the four directions of the intersection selected as a use case. The measurements were taken in the morning and in the afternoon of seven days, collecting 56 videos, and a total of 240 min of recording. This data set was useful to determine the frequency of vehicles at the intersection (to configure the developed simulation) and may be useful for future work related to traffic control in similar cities.

Finally, the developed web application and database allow interaction of users (administrator and TOP) with the TSCS. The interfaces allow an intuitive interaction and are fully integrated with the developed algorithms to allow an adequate traffic control of the intersections managed by the system.

Recommendations for future work related to this research are focused on three aspects. The first aspect is to allow scaling in the flexibility of the developed prototype, in such a way that it is possible to configure the number of directions of an intersection and the number of lanes in each direction. This would allow applying this prototype in a greater number of intersections in a city, with a simpler configuration. The second aspect is to allow more categories for green light times (currently limited to Short Time and Long Time), and describing the possible states in the environment in a greater detail. This improvement could allow obtaining better results in the designed and executed tests. The third aspect is the implementation of a TSCS with a broader scope (more than one intersection) proposed in the same target context. The extension of the scope could be tested with independent agents and with multiple RL agents to evaluate the results obtained in each case. It is important to consider that the development of the complete system and its implementation are proposed, not only the simulation, for which the required hardware and communications components should be considered.

5. Conclusions

The developed prototype for the TSCS was based on a suitable ITS architecture, which considered international reference architectures and the context of a city in a developing country. The prototype allows integration with other mobility systems due to its modular design. This work showed that RL is a suitable approach for this type of situation where decision-based learning is required.

Additionally, the selected components of the prototype, the suggested technologies, and the simplicity of the algorithms should allow easy implementation in cities with similar contexts. Most of the components of the developed TSCS were tested in a real environment, obtaining very good results.

The prototype component in charge of counting the vehicles through computer vision was tested through four phases, obtaining good results. The precision and recall metrics were used to evaluate performance. In the final testing phase of this module, recall was 92.67% and precision was 100%. These results were considered sufficient and adequate, because the TSCS does not require calculating an exact number of vehicles to determine if the vehicle queue is long or short.

The component in charge of making the recommendation time for the traffic signals was also tested. The operation was as expected, and real-time integration tests were performed to guarantee that the response time of the system was consistent with the operation that it would have in a real environment. The response time when requesting a new status of the traffic signals (new times for the phases) was also consistent with the duration of the cycle of the traffic signals at the intersection.

Testing the full functionality of the TSCS, including the traffic signals time modification and measurement of results obtained with this change, would require having an available intersection with traffic signals controlling vehicle flows in all directions. This required scenario was not available for logistics and safety reasons. Therefore, a simulation of the TSCS operation was performed using the SUMO simulation tool. This simulation was tested by comparing it with two other types of traffic control (fixed times in the traffic signals and traffic control with a basic recommendation). The simulation results identified an improvement in the main goal; vehicle queue length was reduced approximately 29% with respect to control with fixed times, and 3% with respect to basic recommendation. Moreover, the simulation identified a reduction of the waiting time and time loss of the vehicles passing through the intersection of approximately 50% with respect to control with fixed times, and of approximately 5% with respect to control with a basic recommendation, with the advantage of low complexity in the development of the system.

Author Contributions: Conceptualization, R.S.-C. and Á.P.d.l.C.; methodology, R.S.-C., Á.P.d.l.C., and J.M.M.M.; software, R.S.-C., J.H.-G., and J.D.R.; validation, R.S.-C., J.H.-G., and J.D.R.; formal analysis, R.S.-C., J.H.-G., J.D.R., Á.P.d.l.C., and J.M.M.M.; investigation, J.H.-G., J.D.R., and R.S.-C.; resources, Á.P.d.l.C. and J.M.M.M.; writing—original draft preparation, R.S.-C., J.H.-G., and J.D.R.; writing—review and editing, Á.P.d.l.C. and J.M.M.M.; supervision, R.S.-C., Á.P.d.l.C., and J.M.M.M.; funding acquisition, Á.P.d.l.C. and J.M.M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Available online at: https://bit.ly/3apf7i6, https://github.com/Julianuto/traffic-lights_recommendation, https://github.com/Julianuto/cross_simulation (accessed on 24 September 2021).

Acknowledgments: The authors wish to thank Universidad del Cauca (Telematics Department) and Universidad Icesi (ICT Department) for supporting this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, J.; Wang, F.Y.; Wang, K.; Lin, W.H.; Xu, X.; Chen, C. Datadriven Intelligent Transportation Systems: A survey. *IEEE Trans. Intell. Transp. Syst.* **2011**, *12*, 1624–1639. [CrossRef]
2. Los Angeles Tops INRIX Global Congestion Ranking—INRIX. Available online: http://inrix.com/press-releases/scorecard-2017/ (accessed on 15 January 2020).
3. Kanungo, A.; Sharma, A.; Singla, C. Smart traffic lights switching and traffic density calculation using video processing. In Proceedings of the Recent Advances in Engineering and Computational Sciences (RAECS), Chandigarh, India, 6–8 March 2014; pp. 1–6. [CrossRef]
4. Zheng, G.; Zang, X.; Xu, N.; Wei, H.; Yu, Z.; Gayah, V.; Xu, K.; Li, Z. Diagnosing Reinforcement Learning for Traffic Signal Control. *arXiv* **2019**, arXiv:1905.04716.
5. Wei, H.; Zheng, G.; Gayah, V.; Li, Z. A Survey on Traffic Signal Control Methods. *arXiv* **2019**, arXiv:1904.08117.
6. Zaatouri, K.; Ezzedine, T. A Self-Adaptive Traffic Light Control System Based on YOLO. In Proceedings of the International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), Hamammet, Tunisia, 20–21 December 2018; pp. 16–19. [CrossRef]
7. Garg, D.; Chli, M.; Vogiatzis, G. Deep reinforcement learning for autonomous traffic light control. In Proceedings of the 3rd IEEE International Conference on Intelligent Transportation Engineering, ICITE, Singapore, 3–5 September 2018; pp. 214–218. [CrossRef]
8. Natafgi, M.B.; Osman, M.; Haidar, A.S.; Hamandi, L. Smart Traffic Light System Using Machine Learning. In Proceedings of the 2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET), Lebanon, Beirut, 14–15 September 2018; pp. 1–6. [CrossRef]
9. Gokulan, B.P.; Srinivasan, D. Distributed geometric fuzzy multi-agent urban traffic signal control. *IEEE Trans. Intell. Transp. Systems* **2010**, *11*, 714–727. [CrossRef]
10. Salman, M.A.; Ozdemir, S.; Celebi, F.V. Fuzzy Traffic Control with Vehicle-to-Everything Communication. *Sensors* **2018**, *18*, 368. [CrossRef]
11. García-Nieto, J.; Albaa, E.; Olivera, A.C. Swarm intelligence for traffic light scheduling: Application to real urban areas. *Eng. Appl. Artif. Intell.* **2012**, *25*, 274–283. [CrossRef]
12. Hu, W.; Wang, H.; Liping, Y.; Du, B. A swarm intelligent method for traffic light scheduling: Application to real urban traffic networks. *Appl. Intell.* **2015**, *44*, 208–231. [CrossRef]
13. Xu, L.H.; Xia, X.H.; Luo, Q. The study of reinforcement learning for traffic self-adaptative control under multiagent Markov game environment. *Math. Problems Eng.* **2013**, *2013*, 962869. [CrossRef]
14. Aslani, M.; Seipel, S.; Wiering, M. Continuous residual reinforcement learning for traffic signal control optimization. *Can. J. Civ. Eng.* **2018**, *45*, 690–702. [CrossRef]
15. Greguríć, M.; Vujić, M.; Alexopoulos, C.; Miletić, M. Application of Deep Reinforcement Learning in Traffic Signal Control: An Overview and Impact of Open Traffic Data. *Appl. Sci.* **2020**, *10*, 4011. [CrossRef]
16. Gu, J.; Fang, Y.; Sheng, Z.; Wen, P. Double Deep Q-Network with a Dual-Agent for Traffic Signal Control. *Appl. Sci.* **2020**, *10*, 1622. [CrossRef]
17. Duowei, L.; Jianping, W.; Ming, X.; Ziheng, W.; Kezhen, H. Adaptive Traffic Signal Control Model on Intersections Based on Deep Reinforcement Learning. *J. Adv. Transp.* **2020**, *2020*, 14. [CrossRef]
18. Liang, X.; Du, X.; Wang, G.; Han, Z. A Deep Reinforcement Learning Network for Traffic Light Cycle Control. *IEEE Trans. Veh. Technol.* **2019**, *68*, 1243–1253. [CrossRef]
19. Lee, J.; Chung, J.; Sohn, K. Reinforcement Learning for Joint Control of Traffic Signals in a Transportation Network. *IEEE Trans. Veh. Technol.* **2020**, *69*, 1375–1387. [CrossRef]

20. Chin, Y.K.; Kow, W.Y.; Khong, W.L.; Tan, M.K.; Teo, K.T.K. Q-Learning Traffic Signal Optimization within Multiple Intersections Traffic Network. In Proceedings of the 2012 Sixth UKSim/AMSS European Symposium on Computer Modeling and Simulation, Valletta, Malta, 14–16 November 2012; pp. 343–348.
21. Belletti, F.; Haziza, D.; Gomes, G.; Bayen, A.M. Expert Level Control of Ramp Metering Based on Multi-Task Deep Reinforcement Learning. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 1198–1207. [CrossRef]
22. Shabestary, S.M.A.; Abdulhai, B. Deep Learning vs. Discrete Reinforcement Learning for Adaptive Traffic Signal Control. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 286–293.
23. Chen, C.; Liu, B.; Wan, S.; Qiao, P.; Pei, Q. An Edge Traffic Flow Detection Scheme Based on Deep Learning in an Intelligent Transportation System. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 1840–1852. [CrossRef]
24. Lopez, P.A.; Wiessner, E.; Behrisch, M.; Bieker-Walz, L.; Erdmann, J.; Flotterod, Y.-P.; Hilbrich, R.; Lucken, L.; Rummel, J.; Wagner, P. Microscopic Traffic Simulation using SUMO. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2575–2582. [CrossRef]
25. Araghi, S.; Khosravi, A.; Creighton, D. A review on computational intelligence methods for controlling traffic signal timing. *Expert Syst. Appl.* **2015**, *42*, 1538–1550. [CrossRef]
26. Aslani, M.; Seipel, S.; Mesgari, M.; Wiering, M. Traffic signal optimization through discrete and continuous reinforcement learning with robustness analysis in downtown Tehran. *Adv. Eng. Inform.* **2018**, *38*, 639–655. [CrossRef]
27. Genders, W.; Razavi, S. Evaluating reinforcement learning state representations for adaptive traffic signal control. *Procedia Comput. Sci.* **2018**, *130*, 26–33. [CrossRef]
28. Pop, M. Traffic Lights Management Using Optimization Tool. *Procedia Soc. Behav. Sci.* **2018**, *238*, 323–330. [CrossRef]
29. García-Ródenas, R.; López-García, M.; Sánchez-Rico, M.; López-Gómez, J. A bilevel approach to enhance prefixed traffic signal optimization. *Eng. Appl. Artif. Intell.* **2019**, *84*, 51–65. [CrossRef]
30. Mannion, P.; Duggan, J.; Howley, E. Parallel Reinforcement Learning for Traffic Signal Control. *Procedia Comput. Sci.* **2015**, *52*, 956–961. [CrossRef]
31. Sanchez-Iborra, R.; Ingles-Romero, J.F.; Domenech-Asensi, G.; Moreno-Cegarra, J.L.; Cano, M. Proactive Intelligent System for Optimizing Traffic Signaling. In Proceedings of the IEEE 14th International Conference on Dependable, Autonomic and Secure Computing, Auckland, New Zealand, 8–12 August 2016; pp. 544–551. [CrossRef]
32. Slimani, N.; Slimani, I.; Sbiti, N.; Amghar, M. Traffic forecasting in Morocco using artificial neural networks. *Procedia Comput. Sci.* **2019**, *151*, 471–476. [CrossRef]
33. Xin, C.; Na, C.; Yeshuai, B. Analysis on Key Technologies of Traffic Prediction and Path Guidance in Intelligent Transportation. In Proceedings of the International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Changsha, China, 17–18 December 2016; pp. 5–8.
34. Wang, Y.; Yang, X.; Liang, H.; Liu, Y. A Review of the Self-Adaptive Traffic Signal Control System Based on Future Traffic Environment. *J. Adv. Transp.* **2018**, *2018*, 1–12. [CrossRef]
35. Yau, K.; Qadir, J.; Khoo, H.; Ling, M.; Komisarczuk, P. A Survey on Reinforcement Learning Models and Algorithms for Traffic Signal Control. *ACM Comput. Surv.* **2017**, *50*, 1–38. [CrossRef]
36. Jin, J.; Ma, X. Hierarchical multi-agent control of traffic lights based on collective learning. *Eng. Appl. Artif. Intell.* **2018**, *68*, 236–248. [CrossRef]
37. Gonzalez, R.A.; Ferro, R.E.; Liberona, D. Government and governance in intelligent cities, smart transportation study case in Bogotá Colombia. *Ain Shams Eng. J.* **2020**, *11*, 25–34. [CrossRef]
38. Chavhan, S.; Venkataram, P. Prediction based traffic management in a metropolitan area. *J. Traffic Transp. Eng. (Engl. Ed.)* **2020**, *7*, 447–466. [CrossRef]
39. Prabha, R.; Kabadi, M.G. KNODET: A Framework to Mine GPS Data for Intelligent Transportation Systems at Traffic Signals. In Proceedings of the International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT), Bangalore, India, 16–17 March 2017; pp. 85–89.
40. Jang, K.; Kim, H.; Jang, I.G. Traffic Signal Optimization for Oversaturated Urban Networks: Queue Growth Equalization. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2121–2128. [CrossRef]
41. Faizan, R.; Kok-Lim, A.Y.; Yeh-Ching, L. Deep reinforcement learning for traffic signal control under disturbances: A case study on Sunway city, Malaysia. *Future Gener. Comput. Syst.* **2020**, *109*, 431–445. [CrossRef]
42. Miletić, M.; Kušić, K.; Gregurić, M.; Ivanjko, E. State Complexity Reduction in Reinforcement Learning based Adaptive Traffic Signal Control. In Proceedings of the 2020 International Symposium ELMAR, Zadar, Croatia, 14–15 September 2020; pp. 61–66. [CrossRef]
43. El-Tantawy, S.; Abdulhai, B. Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC). In Proceedings of the 2012 15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, USA, 16–19 September 2012; pp. 319–326. [CrossRef]
44. Jin, J.; Kalaie, A. A Learning-based Adaptive Signal Control System with Function Approximation. In Proceedings of the 14th IFAC Symposium on Control in Transportation Systems CTS 2016, Istanbul, Turkey, 18–20 May 2016.
45. Salazar-Cabrera, R.; Pachón, A.; Madrid, J.M. Proof of Concept of an IoT-Based Public Vehicle Tracking System, Using LoRa (Long Range) and Intelligent Transportation System (ITS) Services. *J. Comput. Netw. Commun. Hindawi* **2019**, *2019*, 9198157. [CrossRef]

46. Architecture Reference for Cooperative and Intelligent Transportation. Available online: https://local.iteris.com/arc-it/ (accessed on 18 January 2020).
47. Relationship with the ITS Action Plan and ITS Directive. FRAME ARCHITECTURE. Available online: https://frame-online.eu/frame-architecture/detailed-information/relationship-with-the-its-action-plan-and-its-directive (accessed on 11 December 2020).
48. ISO 14813-1:2015(en). Intelligent Transport Systems—Reference Model Architecture(s) for the ITS Sector—Part 1: ITS Service Domains, Service Groups and Services. Available online: https://www.iso.org/obp/ui/#iso:std:iso:14813:-1:ed-2:v1:en (accessed on 15 December 2020).
49. Arquitectura Nacional ITS de Colombia. Available online: http://www.consystec.com/colombia/web/ (accessed on 22 January 2020).
50. Act 1450 of 2011, Article 84; Decree 2060 of October 2015. Available online: http://wp.presidencia.gov.co/sitios/normativa/decretos/2015/Decretos2015/DECRETO%202060%20DEL%2022%20DE%20OCTUBRE%20DE%202015.pdf (accessed on 20 August 2021).
51. The Browsing Tool. FRAME ARCHITECTURE. Available online: https://frame-online.eu/frame-architecture/the-browsing-tool (accessed on 9 December 2020).
52. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
53. Welcome to Flask—Flask Documentation (1.1.x). Available online: https://flask.palletsprojects.com/en/1.1.x/ (accessed on 16 December 2020).
54. Ngrok. Available online: https://ngrok.com/ (accessed on 23 January 2020).
55. Gym. Available online: https://gym.openai.com/ (accessed on 22 January 2020).
56. Gym. Taxi Environment Example. Available online: https://gym.openai.com/envs/Taxi-v3/ (accessed on 18 December 2020).
57. Reinforcement Q-Learning from Scratch in Python with OpenAI Gym—LearnDataSci. Available online: https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/ (accessed on 13 December 2020).
58. Traffic Lights Recommendation Algorithm. GitHub Link. Available online: https://github.com/Julianuto/traffic-lights_recommendation (accessed on 20 August 2021).
59. Traffic Lights Recommendation Algorithm with SUMO Simulation. GitHub Link. Available online: https://github.com/Julianuto/cross_simulation (accessed on 20 August 2021).
60. TraCI-SUMO Documentation. Available online: https://sumo.dlr.de/docs/TraCI.html (accessed on 13 December 2020).
61. Caraffi, C.; Vojíř, T.; Trefný, J.; Šochman, J.; Matas, J. A system for real-time detection and tracking of vehicles from a single car-mounted camera. In Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, USA, 16–19 September 2012; pp. 975–982. [CrossRef]
62. Cao, C.-Y.; Zheng, J.-C.; Huang, Y.-Q.; Liu, J.; Yang, C.-F. Investigation of a Promoted You Only Look Once Algorithm and Its Application in Traffic Flow Monitoring. *Appl. Sci.* **2019**, *9*, 3619. [CrossRef]