الجمهورية الشعبية الديمقراطية الجزائرية
People's Democratic Republic of Algeria
وزارة التعليم العالي و البحث العلمي
Ministry of Higher Education and Scientific Research
جامعة ابن خلدون - تيارت
University of Ibn Khaldoun - Tiaret

Master's Thesis

To obtain the diploma of **Master's Degree**

Field of Study: **Electrical Engineering**

Specialization: **Electronics and Embedded Systems**

Theme

Intelligent Traffic Management System Based on AI and IoT

Presented by
ABDI Souhila Fatima

Defended on: **July, 2025**
In front of the jury composed of

Mr. Youcef BELHADJI	President of the Jury
Mr. Ahmed SAFA	Thesis Supervisor
Mr. Mustapha BELARBI	Co-Supervisor
Mr. Noureddine MAAMAR	Examiner

Academic Year: 2024/2025

Acknowledgment

First and foremost, I thank Allah, the Most Gracious and the Most Merciful,for granting me the strength, patience, and wisdom to complete this work.

I would like to express my deepest gratitude to my beloved parents and my siblings, for their unconditional love, support, and continuous encouragement throughout my academic journey.

A heartfelt thanks to my brother-in-law, ***Abdulsamd****, for his invaluable motivation and constant encouragement.*

I am sincerely grateful to my supervisor, ***Dr. Safa Ahmed****, for his guidance, and support throughout the realization of this project.*

My warmest thanks also go to my dearest friends, ***Hiba****,* ***Mimi****, and* ***Douaa****, for their unwavering friendship and encouragement, and to everyone who contributed to the successful completion of this work in any way.*

To all of you, thank you from the bottom of my heart.

Dedication

I dedicate this work to my beloved parents, whose unconditional love, sacrifices, and encouragement have shaped who I am today.

To my dear siblings, thank you for always being by my side, for your patience, and for your constant support.

This achievement is as much yours as it is mine.

Abstract

The high number of vehicles on the roads today poses a serious hurdle with respect to traffic management on the roads, particularly considering the worsening and narrowed state of Tiaret's road infrastructure. Inflexible traffic signal scheduling at every intersection complicates traffic congestion, environmental deterioration, excessive fuel usage, and driver dissatisfaction. The proposed solution for this problem includes an artificial intelligence and Internet of Things-based system that operates on the premise of real-time processing of data.

Traffic congestion within metropolitan areas is a chronic problem that results in traffic bottlenecks, environmental deterioration, and issues of ensuring road safety. This study assumes the implementation and development of a holistic traffic management system based on Artificial Intelligence and the Internet of Things to streamline traffic movement and enhance road safety. Under the suggested framework, the detection and enumeration of vehicles and pedestrians are achieved using real-time video input along with object identification techniques. The observations are then analyzed to control traffic signal duration according to usage levels of roads, pedestrian requirements, and emergency vehicle prioritization. The Raspberry Pi 5 is used as the computational unit to implement this framework practically. It is expected to enhance traffic functioning efficiency and response times while being a cost-effective and efficient solution to deploy within smart city projects.

Keywords— Traffic management, Artificial Intelligence, IoT, Traffic lights, Smart city, Vehicle detection

List of Acronyms

AI	Artificial Intelligence
IoT	Internet of Things
YOLO	You Only Look Once
RL	Reinforcement Learning
CNN	Convolutional Neural Network
DNN	Deep Neural Network
ITS	Intelligent Transportation Systems
ATC	Advanced Transportation Controller
NTCIP	National Transportation Communications for ITS Protocol
C-ITS	Cooperative Intelligent Transportation Systems
V2I	vehicle-to-infrastructure communication
TLA	Traffic Light Assistance
RFID	Radio-frequency identification
EVP	Emergency Vehicle Preemption
CI	Computational intelligence
GA	genetic algorithms
SA	simulated annealing
TST	traffic signal timing
MBBNet	The Multi-BackBone Network
MQTT	Message Queuing Telemetry Transport
A2C	Advantage Actor-Critic
DQN	Deep Q-Network
Q-Learning	Quality Learning
OpenCV	Open Source Computer Vision Library
SSD	Single Shot Multibox Detector
FRCNN	Faster Region-based Convolutional Neural Network
FPS	Frames Per Second
mAP	Mean Average Precision
IoU	Intersection over Union
SUMO	Simulation of Urban MObility
ID	Identifier
CPU	Central Processing Unit
GPU	Graphics Processing Unit
CSV	Comma-Separated Values
ML	Machine Learning
PR Curve	Precision-Recall Curve
F1-Score	Harmonic Mean of Precision and Recall

Contents

List of Figures

General Introduction

Traffic congestion is a pervasive challenge in modern cities, imposing severe socio-economic and environmental costs. Urban congestion not only wastes commuters' time and fuel, but also drives up CO_2 emissions and reduces transportation efficiency [1]. Many cities continue to operate traffic lights on fixed schedules (fixed-time or actuated controllers), which are not responsive to real-time demand. As noted by Damadam *et al.*, fixed-time signals "are not optimal because they are not affected by the traffic conditions, and traffic changes over time" [2]. In practice, this inflexibility leads to unnecessary delays and queues during peak traffic periods [2, 3].

Computer vision and object detection are key technologies for real-time traffic analysis. Contemporary deep-learning detectors (e.g., YOLOv8, Keras) can identify and count vehicles in camera feeds at high speed. These vision models significantly improve detection accuracy in complex scenes, enabling robust real-time monitoring of intersections [4]. In fact, state-of-the-art target-detection networks have become widely applied to traffic surveillance, since accurately identifying vehicles and pedestrians enhances the intelligence and automation of city traffic management [4].

Modern adaptive signal control leverages AI algorithms together with IoT sensor data. IoT devices (such as cameras) continuously collect real-time data on traffic flow, density, and queue lengths [3]. Machine learning models analyze these data streams to detect patterns and predict congestion in advance [3]. For example, Damadam *et al.* applied surveillance cameras and a distributed multi-agent deep reinforcement learning algorithm across multiple intersections; their system shared local and neighboring traffic data to learn optimal signal timing policies in real time [2]. In simulations, this AI-based approach reduced average queue lengths and waiting times compared to the city's existing fixed-time control [2].

The objective of this study is to design and evaluate an AI- and IoT-based smart traffic light management system that uses real-time object detection to adapt signal timings based on the density of vehicles on the road and emergency vehicle existence. We ask: *how can vision-based vehicle counting and IoT-enabled communication be integrated to optimize traffic flow and reduce congestion?* To answer this, we develop a system that combines a deep learning object detector (for counting vehicles from camera feeds) with an IoT network linking multiple intersections. The system uses machine learning to adjust green-phase durations dynamically in response to current conditions. We hypothesize that this adaptive approach will significantly lower delays and queue lengths relative to conventional fixed-time signals. The remainder of this thesis is organized as follows.

Chapter 1 reviews the **state of the art** in traffic light systems, charting the shift from conventional fixed-time systems to AI-based intelligent approaches and ends with a case study on traffic challenges in Tiaret, Algeria.

Chapter 2 elaborates on the creation of the **object detection system** using YOLOv8

and OpenCV, covering model training, performance metrics, and its implementation on Raspberry Pi.

Chapter 3 investigates the application of **reinforcement learning algorithms** like DQN and A2C for optimizing traffic signals, with discussions on their design, performance in SUMO simulations, and learning efficacy.

Chapter 4 offers a detailed **evaluation of the proposed smart traffic management system**, contrasting traditional and intelligent control systems in both simulated and real-world scenarios.

Finally, the **conclusion** distills the main findings, addresses project constraints, and suggests paths for future enhancements and practical deployment.

State of the Art of Traffic Light Systems

Traffic management has emerged as one of the foremost issues in contemporary urban settings. The surge in vehicle numbers, paired with antiquated traffic control systems, has resulted in escalated traffic jams, increased fuel usage, and heightened environmental pollution. Central to the regulation of urban traffic are traffic light systems, which are vital for the coordination of vehicles and pedestrians at crossroads.
This chapter presents a detailed examination of the current advancements in traffic light systems, beginning with their core components and operational roles. It explores the historical progression of these systems, followed by an analysis of the technological innovations that have altered traffic signal management over time. In the latter part of the chapter, recent research trajectories are explored, including the incorporation of Artificial Intelligence (AI), Deep Learning, Reinforcement Learning, and Internet of Things (IoT) technologies into traffic management solutions. These state-of-the-art approaches seek to overcome the drawbacks of traditional fixed-time systems by introducing real-time adaptability, forecasting abilities, and prioritization of emergency services. Finally, the chapter delves into the particular context of traffic congestion in Algeria, focusing on Tiaret city and the Regina intersection as a local case study. This section underscores the necessity for intelligent traffic light systems customized to the specific conditions of Algerian urban areas.

1 Definition and Function of Traffic Light Systems

Traffic signals are automated devices installed at intersections and other key points on roads to control the movement of vehicles, bicycles, and pedestrians. They use colored lights—red, yellow, and green—to assign right-of-way and regulate the flow of traffic. The primary goal of traffic signals is to enhance safety and improve the efficiency of road networks by reducing conflicts between different types of road users.[5]

1.1 Key Components of Traffic Signals

Traffic signals are composed of several components that work together to manage traffic flow. Engineers must design these components for reliability, visibility, and adaptability to changing traffic conditions.[5]

1.1.1 Signal Heads

Signal heads are the most visible part of a traffic signal, displaying red, yellow, and green lights to control the movement of vehicles and pedestrians. These lights must be bright, visible in all weather conditions, and positioned to be easily seen by drivers.[5]

1.1.2 Controllers

The controller is the brain of the traffic signal system. It regulates the timing and sequence of the lights, adjusting based on traffic flow and patterns.[5]

1.1.3 Detectors

Detectors are sensors that detect the presence of vehicles, bicycles, or pedestrians at intersections. These sensors feed data to the controller, which then adjusts the signal timing based on the volume of traffic.[5]

1.1.4 Pedestrian Signals

Pedestrian signals provide specific instructions for pedestrians, typically with “Walk” and “Don’t Walk” indicators.[5]

2 Historical Development of Traffic Signals

The first traffic signal was installed in London on December 10, 1868. It was gas-lit and manually operated by a police officer using semaphore arms during the day and red and green lights at night.[6]Unfortunately, this early system was short-lived, as it exploded due to a gas leak, injuring the officer operating it.[7]

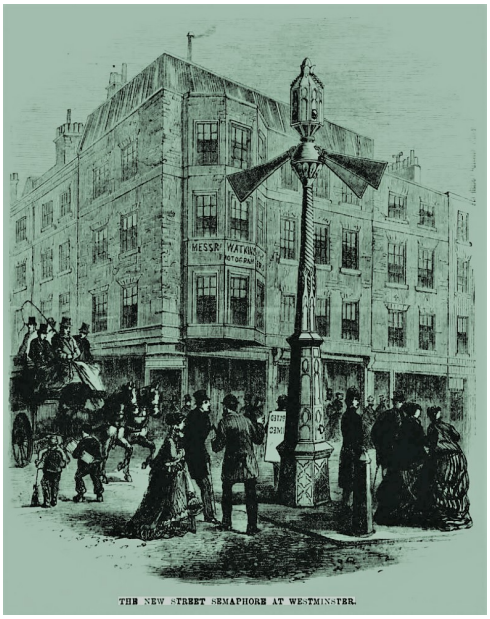


Figure 1: The first traffic light in London [8]

The first electric traffic light was introduced in 1912 by Lester Wire, a police officer in Salt Lake City, USA.[6] By 1920, the modern three-color system (red, yellow, green)

was developed in Detroit, revolutionizing traffic control.[9]

Figure 2: The first traffic light in the USA [10]

3 Evolution and Technological Advancements

The evolution of traffic light systems reflects both technological innovation and the growing complexity of urban traffic. Early systems relied on manual operation or fixed-time schedules, which were inadequate for fluctuating traffic patterns. The 1950s marked a shift with the introduction of actuated signals, using loop detectors to adjust timings based on vehicle presence, a significant improvement over clockwork mechanisms[11]
In the late 1990s, the Advanced Transportation Controller (ATC) initiative in the United States aimed to standardize traffic light controllers under the National Intelligent Transportation System program. This led to the adoption of the National Transportation Communications for ITS Protocol (NTCIP), enabling controllers to communicate using Internet Protocol and improving coordination across networks.[12]
Modern traffic lights leverage Cooperative Intelligent Transportation Systems (C-ITS), which enable vehicle-to-infrastructure (V2I) communication. For example, Traffic Light Assistance (TLA) systems calculate optimal speeds for vehicles approaching intersections, reducing stops and emissions. A 2018 study demonstrated that TLA systems using V2I communication improved traffic flow and reduced waiting times in simulated environments[13]
Smart traffic lights, emerging in the 2010s, integrate AI, sensors, and machine learning to adapt dynamically to traffic conditions. Systems like New York City's Midtown in Motion use cameras, microwave sensors, and RFID E-ZPass readers to monitor traffic and adjust

signals in real-time, reducing delays by up to 40% in some cities like Pittsburgh[12, 14]. Emergency Vehicle Preemption (EVP) systems prioritize first responders by adjusting signals, minimizing delays and enhancing safety[15].

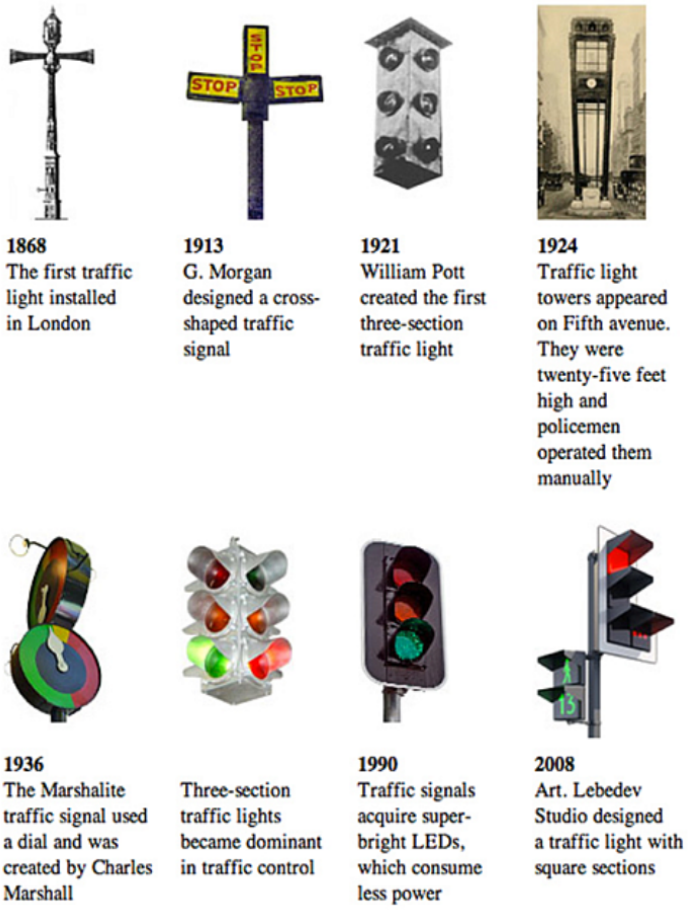


Figure 3: Developpement of traffic light systems [16].

4 Current Research in Traffic Light Systems

4.1 Computational Intelligence for Traffic Signal Timing Optimization

Computational intelligence (CI) techniques, such as genetic algorithms (GA), simulated annealing (SA), and fuzzy logic, have been extensively applied to traffic signal timing (TST) optimization. A 2020 review highlighted the effectiveness of CI in addressing the stochastic nature of traffic, particularly at intersections. For instance, GA-based methods iteratively evolve signal timing plans to minimize delays, often outperforming traditional fixed-time systems by 15–20% in simulated urban scenarios[17]. Fuzzy logic, on the other hand, uses rule-based systems to adjust signal phases based on traffic density and queue

lengths, offering robustness in handling uncertainty[18].

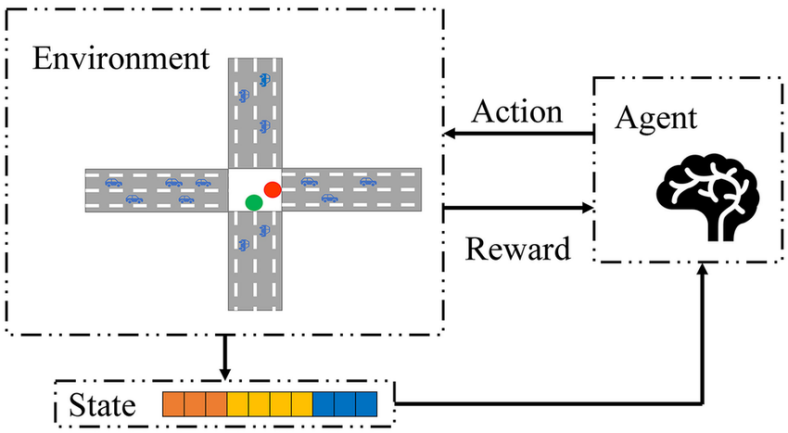


Figure 4: Reinforcement learning structure in traffic signal control system, which consists of a DRL agent, a traffic environment, and three signals: reward, state, and action [19].

4.2 Reinforcement Learning for Adaptive Traffic Control

Reinforcement learning (RL) has emerged as a powerful approach for adaptive TSC, enabling controllers to learn optimal signal policies through trial-and-error interactions with traffic environments. RL-based multi-agent systems, where each intersection operates as an independent agent, have shown promise in reducing delays. A foundational study by Arel et al. (2010) applied Q-learning to network traffic signal control, achieving smoother flow compared to fixed-time systems in a simulated grid network[20]. More recently, deep RL methods like Deep Deterministic Policy Gradient (DDPG) and Deep Q-Networks (DQN) have been explored for their ability to handle continuous action spaces and complex traffic dynamics.

4.3 Deep Learning for Traffic Detection and Prediction

Deep learning (DL) techniques are transforming traffic light systems, particularly in detection and prediction tasks. The Multi-BackBone Network (MBBNet), a lightweight deep convolutional neural network (CNN), was proposed for traffic light detection in autonomous vehicles, balancing accuracy and computational efficiency for edge devices. MBBNet achieved a detection accuracy of 92% while maintaining low latency, making it suitable for real-time applications[21]. Another study by Abbas (2024) utilized Faster R-CNN for vehicle detection at intersections, enabling dynamic signal adjustments with a reported accuracy of 96.6% in real-world scenarios[22]. Traffic flow prediction, a critical component for proactive TSC, has also benefited from DL. A 2023 review emphasized the integration of CNNs and Long Short-Term Memory (LSTM) networks in Intelligent Transportation Systems (ITS) to forecast traffic patterns. These models enable traffic

lights to anticipate congestion and adjust timings preemptively, reducing delays by up to 10% in simulated tests. However, the review noted challenges in obtaining sufficient data from local authorities, particularly for smaller cities, limiting widespread adoption[23].

4.4 Emerging Approaches and Hybrid Systems

Emerging research explores hybrid systems that combine rule-based methods with AI to address the limitations of standalone RL or DL approaches. For example, the Self-Organizing Traffic Lights (SOTL) method, a rule-based decentralized approach, has been integrated with RL to improve adaptability while maintaining stability.[24].

4.5 IoT in Traffic Management

The integration of the Internet of Things (IoT) has emerged as a transformative paradigm in intelligent transportation systems, enabling real-time data collection, communication, and control to address urban traffic challenges. IoT encompasses a network of interconnected devices—sensors, actuators, and edge computing units—that facilitate the seamless exchange of information, supporting adaptive traffic management solutions[25].
Early applications of IoT in traffic management focused on static sensor networks, such as inductive loops and radar, to monitor vehicle presence and inform signal timings [26]. However, recent advancements have shifted toward edge computing devices, such as Raspberry Pi and Arduino platforms, which process data locally and transmit it wirelessly to centralized systems or simulations like SUMO. Studies have demonstrated that IoT-enabled systems can reduce traffic delays by up to 15–25% through real-time vehicle detection and prioritization, particularly for emergency vehicles [27]. For instance, vision-based IoT solutions using cameras and machine learning have been proposed to detect traffic conditions, offering a scalable alternative to traditional infrastructure[22].

4.6 Challenges in Current Research

Current research in traffic light systems encounters several significant hurdles that impact their development and deployment. Reinforcement learning (RL) approaches, such as Deep Deterministic Policy Gradient (DDPG) and Deep Q-Networks (DQN), require substantial computational resources and extensive training datasets, often leading to inconsistent performance across varying traffic conditions [28]. The lack of standardized and comprehensive traffic datasets poses a further obstacle, particularly for deep learning (DL) models used in traffic flow prediction, where insufficient data from local authorities limits scalability, especially in smaller urban areas [23]. Additionally, the deployment of Cooperative Intelligent Transportation Systems (C-ITS) and vehicle-to-infrastructure (V2I)

communication introduces cybersecurity risks, as connected infrastructure remains vulnerable to potential attacks that could disrupt traffic flow or compromise safety [27]. The integration of hybrid systems combining rule-based methods with artificial intelligence also faces difficulties in balancing adaptability and stability, complicating their real-world implementation [24]. Moreover, the computational demands of advanced models like the Multi-BackBone Network (MBBNet) and Faster R-CNN for real-time detection challenge their feasibility on edge devices with limited processing power [29, 21]. The integration of Internet of Things (IoT) technologies, such as edge computing and wireless communication, introduces additional considerations.

5 Traffic Jams in Algeria

5.1 Traffic Congestion in Algeria

Algeria, as a North African nation with a rapidly growing urban population, faces significant challenges with traffic congestion, particularly in its major cities. The country's road network, spanning approximately 113,655 km, includes 73,607 km of paved roads, supporting a vehicle fleet that grew from 6.2 million in 2016 to an estimated 8 million by 2025[30]. Urban centers like Algiers, Oran, and Constantine experience chronic traffic jams due to population growth, inadequate infrastructure, and increasing vehicle ownership. A 2023 report highlighted that Algiers alone loses an estimated 40 hours per driver annually to congestion, costing the economy millions in fuel and productivity losses[31]. The reliance on fixed-time traffic signals, inherited from mid-20th-century designs, exacerbates congestion in Algerian cities. These systems fail to adapt to dynamic traffic patterns, such as peak hours or unexpected events like festivals, leading to prolonged delays. Recent studies suggest that smart traffic light systems, leveraging real-time data and artificial intelligence, could reduce travel times by up to 30% in similar urban contexts, though implementation remains limited due to funding and technical expertise shortages[32].

(a) Traffic jam in Oran city[33]

(b) Traffic jam in Algiers city[34]

Figure 5: Overview of some congested roads in Algeria

5.2 Traffic Jams in Tiaret City

Tiaret, a mid-sized city in western Algeria with a population of approximately 250,000, mirrors these national challenges with its own localized traffic issues. Located 160 km southwest of Algiers, Tiaret serves as a regional hub, connecting rural areas to urban markets, which increases traffic volume through its central districts. The city's road network, characterized by narrow streets and a mix of modern and historical infrastructure, struggles to accommodate growing vehicle numbers, estimated at 15,000–20,000 vehicles by 2025[35]. Key congestion points include the Regina area, a commercial and residential hub with multiple intersecting roads, where traffic bottlenecks are frequent due to insufficient signal coordination and high pedestrian activity.

Local reports indicate that peak traffic hours (7–9 AM and 4–6 PM) see travel times exceeding 20–30 minutes for short distances (e.g., 2–3 km), reflecting inefficiencies in the current traffic light system[36]. The reliance on manual adjustments by traffic police, rather than automated or adaptive signals, contributes to these delays. Moreover, the city's topography, with its hilly terrain, complicates traffic flow, requiring more sophisticated control strategies to manage uphill and downhill traffic interactions.

5.3 Congested Area in Tiaret (case study : Regina)

The provided Google Maps image (Figure 7) depicts a congested intersection in the Regina area of Tiaret, offering a visual representation of the traffic jam challenges. The image reveals a complex urban layout with multiple intersecting roads surrounding a central roundabout. Dense building coverage and narrow streets suggest limited space for traffic expansion, while visible vehicle clustering indicates heavy congestion, particularly during

peak hours. This intersection likely serves as a critical node, connecting residential zones to commercial areas, amplifying traffic pressure.

Figure 6: Regina, Tiaret[37]

The congestion in Regina can be attributed to several factors observable in the image:

- Intersection Design: The roundabout or multi-road junction lacks modern signal infrastructure.
- Urban Density: The tightly packed buildings indicate a high population density, increasing vehicle and pedestrian volumes.
- Lack of Coordination: The absence of synchronized traffic lights, as inferred from the static traffic pattern, suggests reliance on fixed-time controls, which are inadequate for managing dynamic flows.

6 Conclusion

This chapter has followed the evolutionary route of traffic light systems from manual signaling's start to systems using vehicle-to-infrastructure (V2I) communication plus computational intelligence, reinforcement learning (RL), and deep learning (DL). This progression underscores the fact that we transition from a static time-of-day control to an adaptive data-driven set of methodologies which can potentially improve traffic efficiency and overall safety. As has been reviewed in recent research, these advancements underscore how they have transformative potential, but it identifies hurdles that are meaningful, like high computational demands, availability that is limited for traffic datasets that are standardized, and also cybersecurity vulnerabilities in systems that are connected. From Internet of Things (IoT) technologies' integration, challenges like latency, power constraints, and interoperability emerge. These challenges appear in Tiaret, Algeria's

Regina district, where traffic congestion occurs due to outdated signal systems, narrow street layouts, and rapid urbanization. This local context highlights above the imperative for revolutionary, context-specific solutions. These solutions address technological with environmental constraints. These perceptions do collectively illuminate the nature of multidimensional traffic signal management. Chapter 2 will explore the object detection system because it uses IoT for real-time data collection, as Chapter 3 will examine reinforcement learning strategies as it builds toward the evaluation of the proposed smart traffic management system in Chapter 4.

Object Detection for Smart Traffic Systems

The innovation of advanced traffic management systems has reshaped city mobility through the application of cutting-edge technologies to solve the rising problem of congestion, safety, and operational efficiency of transport. At the core of these innovations is the ability of object detection, which makes real-time observation and evaluation of traffic patterns, hence providing the determining data needed to dynamically manage traffic signals. The chapter addresses the application of object detection in the context of Regina, Tiaret, which is a highly populated urban area located in western Algeria, distinguished by narrow roads, high levels of walking activity, and increasing numbers of motor vehicles, as depicted in the case study in Chapter 1 (Google Maps, 2025). The main focus of this chapter is to clarify the object detection system's design and implementation specifically suited to intelligent traffic applications, using the YOLO (You Only Look Once) framework alongside the OpenCV library. Implemented on the Raspberry Pi 5, this system allows real-time vehicle detection and enumeration, including detecting emergency vehicles, hence supporting the adaptive optimization of traffic signal control. The chapter builds on the aforedescribed hardware setup in relation to the Raspberry Pi 5 with a specific emphasis on its IoT-enabled architecture, and the camera module to the software structure supporting the detection mechanism. The chapter concludes by illustrating how the data produced aligns with the reinforcement learning approaches presented in Chapter 3, hence providing ground for the evaluation in Chapter 4. Subsequent sections present a short theory framework, inspect the hardware setup, and evaluate the implementation using YOLO and OpenCV.

1 Theoretical Background of Object Detection

Object detection is a fundamental task in computer vision that involves identifying and locating objects within images or videos[38]. It combines object detection, a cornerstone of computer vision, is the process of identifying and localizing multiple objects within images or video streams, assigning them specific class labels and spatial coordinates. This task extends beyond image classification, which assigns a single label to an entire image, and semantic segmentation, which assigns labels to every pixel, by providing both classification and precise localization through bounding boxes. The significance of object detection in smart traffic systems lies in its ability to provide real-time, granular data on traffic elements—such as vehicles, pedestrians, and emergency vehicles—enabling dynamic traffic management and safety enhancements. This section traces the evolution of object detection techniques, explores key methodologies, and discusses their relevance and challenges within the context of urban traffic monitoring, particularly for the Regina, Tiaret intersection.

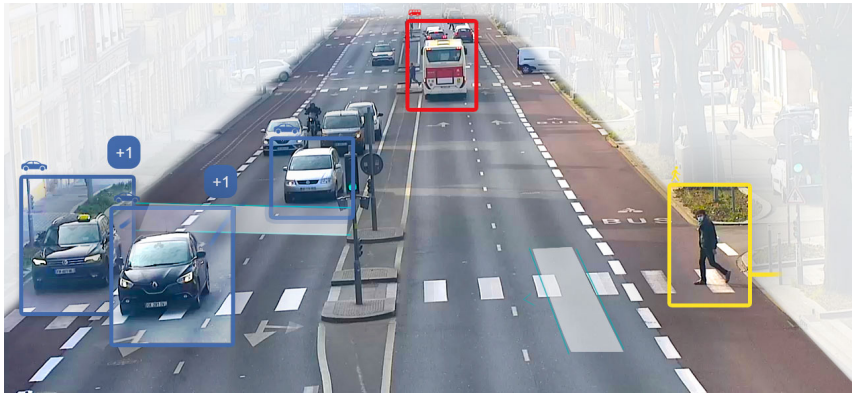


Figure 7: Example of vehicle counting using a smart traffic system [39].

1.1 Evolution of Object Detection Techniques

The development of object detection has undergone significant transformation over the decades. Early methods relied on hand-crafted features and traditional machine learning algorithms. For instance, the Viola-Jones framework, introduced in 2001, utilized Haar-like features and AdaBoost classifiers to detect faces in real-time, laying the groundwork for object detection in constrained settings[40]. Subsequently, the Histogram of Oriented Gradients (HOG) combined with Support Vector Machines (SVM) emerged as a robust approach for detecting objects like pedestrians, leveraging edge orientation histograms to capture shape information[41]. These methods, while effective for specific tasks, were limited by their reliance on manually engineered features, which struggled to generalize across diverse scenarios.

The advent of deep learning marked a paradigm shift, with Convolutional Neural Networks (CNNs) enabling automated feature extraction. The Region-based Convolutional Neural Network (R-CNN), proposed by Girshick et al. (2014), introduced a two-stage process: region proposal followed by classification, significantly improving accuracy but at the cost of computational efficiency. This was followed by Faster R-CNN, which integrated a Region Proposal Network (RPN) to enhance speed, and Mask R-CNN, which added instance segmentation capabilities[42, 43].

However, these multi-stage methods remained computationally intensive, prompting the development of single-stage detectors. The You Only Look Once (YOLO) framework, first introduced by Redmon et al. (2016), revolutionized the field by using a single CNN to predict bounding boxes and class probabilities directly from full images, achieving real-time performance. Subsequent iterations, such as YOLOv3 and YOLOv8, have further refined accuracy and speed, making them suitable for resource-constrained environments like edge devices[44, 45].

1.2 Key Methodologies and Architectures

Object detection methodologies can be broadly categorized into two-stage and single-stage detectors. Two-stage detectors, such as R-CNN and its variants, first generate region proposals and then classify them, offering high precision but requiring significant computational resources. Single-stage detectors, including YOLO and Single Shot Multi-Box Detector (SSD), bypass the proposal step, predicting classes and bounding boxes in one pass, which prioritizes speed over precision in some cases[46]. The YOLO framework's evolution includes enhancements such as the introduction of Darknet-53 as a backbone in YOLOv3, improving feature extraction, and the incorporation of Cross-Stage Partial connections in YOLOv5 to boost performance on small objects[47]. YOLOv8, the latest iteration, further optimizes these aspects with advanced loss functions and post-processing techniques, making it adaptable to diverse object detection tasks[45]. These advancements are particularly relevant for traffic systems, where detecting small or partially occluded objects (e.g., motorcycles, pedestrians) is essential in urban settings like Regina, Tiaret.

1.3 Relevance to Smart Traffic Systems

In the context of smart traffic systems, object detection serves as a critical enabler by providing actionable data for traffic signal optimization, incident detection, and safety management. The ability to count vehicles, classify types (e.g., cars, trucks, emergency vehicles), and detect anomalies (e.g., accidents, jaywalking) in real-time supports adaptive control strategies, reducing congestion and improving response times. For the Regina, Tiaret intersection, where narrow streets and high pedestrian activity exacerbate congestion, object detection can identify traffic bottlenecks and prioritize emergency vehicles, addressing the safety concerns highlighted in Chapter 1.

Figure 8: Illustration of a smart traffic system[48]

2 Data Collection and Object Detection

Effective traffic signal optimization hinges on the availability of accurate, real-time traffic data to inform dynamic control decisions. In the context of the Regina, Tiaret intersection, characterized by high congestion, dense urban surroundings, and significant pedestrian activity, this research employs a vision-based approach for data collection, leveraging machine learning techniques to monitor traffic conditions. Specifically, the methodology integrates the YOLO framework and OpenCV library to perform object detection and vehicle counting, including the identification of emergency vehicles, using a camera interfaced with a Raspberry Pi 5. This section details the hardware setup, software implementation, detection process, and data preprocessing, ensuring a robust foundation for subsequent simulation and control phases.

2.1 Hardware Setup for Real-Time Data Collection

The data collection system is designed around the Raspberry Pi 5, a compact single-board computer selected for its balance of computational power, affordability, and compatibility with IoT applications, making it suitable for deployment in a resource-constrained environment like Tiaret[27].

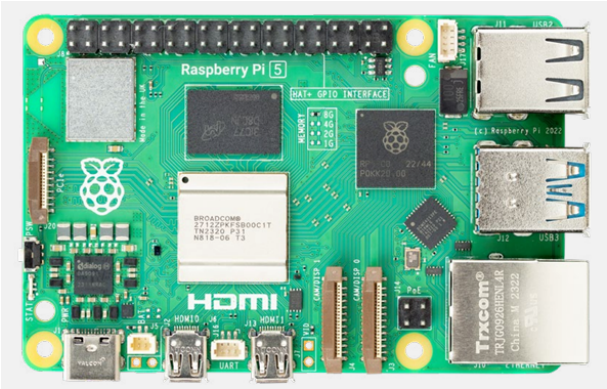


Figure 9: Raspberry Pi5 development board[49]

2.1.1 The Raspberry Pi 5 features

The following specifications are taken from the official Raspberry Pi 5 Product Brief [49]:

- broadcom BCM2712 2.4GHz quad-core 64-bit Arm Cortex-A76 CPU, with cryptography extensions, 512KB per-core L2 caches, and a 2MB shared L3 cache.
- VideoCore VII GPU, supporting OpenGL ES 3.1, Vulkan 1.2
- Dual 4Kp60 HDMI® display output with HDR support

- 4Kp60 HEVC decoder
- LPDDR4X-4267 SDRAM
- (4GB and 8GB SKUs available at launch)
- Dual-band 802.11ac Wi-Fi®
- Bluetooth 5.0 / Bluetooth Low Energy (BLE)
- microSD card slot, with support for high-speed SDR104 mode
- 2 × USB 3.0 ports, supporting simultaneous 5Gbps operation
- 2 × USB 2.0 ports
- Gigabit Ethernet, with PoE+ support (requires separate PoE+ HAT)
- 2 × 4-lane MIPI camera/display transceivers
- PCIe 2.0 x1 interface for fast peripherals (requires separate M.2 HAT or other adapter)
- 5V/5A DC power via USB-C, with Power Delivery support
- Raspberry Pi standard 40-pin header
- Real-time clock (RTC), powered from external battery
- Power button

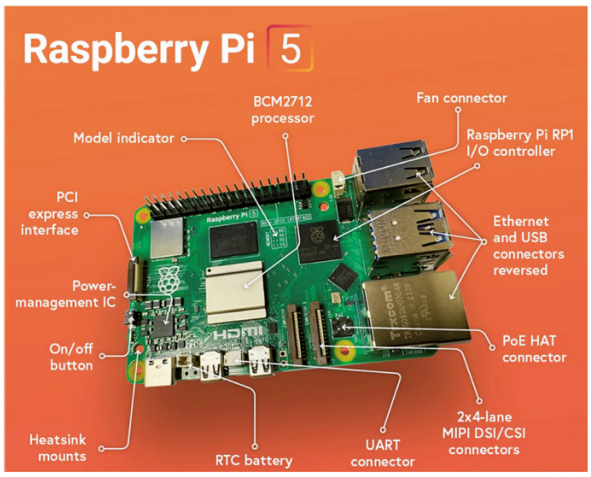


Figure 10: Labelled components for Raspberry Pi 5[49]

2.1.2 IoT Architecture

The object detection system is underpinned by an Internet of Things (IoT) architecture, which facilitates real-time data collection and edge processing. This subsection details the IoT framework, hardware components, and their adaptation to the local context.
IoT Framework:

- Device Role: The Raspberry Pi 5 serves as an IoT edge device, performing on-site object detection and preprocessing, reducing dependence on cloud computing in Tiaret's resource-constrained environment [27].
- Connectivity: Utilizes dual-band Wi-Fi (5 GHz) for data transmission to a local server, with a latency of 10–20 ms, logged on a 64GB microSD card for offline analysis [50].
- Protocols: Employs MQTT for lightweight, reliable data exchange, aligning with IoT standards for traffic applications [25].

2.1.3 Raspberry Pi HQ Camera Features

The Raspberry Pi High Quality (HQ) Camera is designed for professional and hobbyist applications requiring high-resolution imaging. Below are the key features that were taken from the official camera guide for Raspberry Pi[51]:

- **Sensor:** Sony IMX477R stacked, back-illuminated sensor.
- **Resolution:** 12.3 megapixels (4056 × 3040).
- **Lens support:** Compatible with C- and CS-mount lenses.
- **Mount:** Tripod mount with back focus adjustment ring.
- **Shutter:** Rolling shutter.
- **Video modes:** Supports Full HD (1920 × 1080) at up to 120fps (lower resolution).
- **Connection:** 20 cm ribbon cable with MIPI CSI interface.
- **Manual control:** Adjustable aperture and focus.
- **Compatible with:** Raspberry Pi 5, 4, 3, and 2 (requires appropriate software and setup).

The HQ Camera is especially useful for computer vision, surveillance, and scientific projects where image quality and lens flexibility are important.

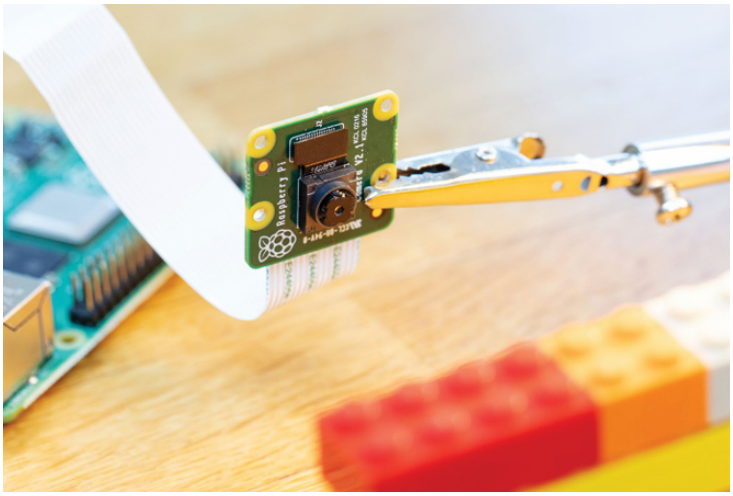


Figure 11: Raspberry Pi camera[51]

The camera is mounted at the Regina intersection to provide a top-down or angled view of the traffic flow, capturing all approaching lanes. The mounting height and angle are adjusted to minimize occlusions caused by vehicles or pedestrians, with an estimated height of 4–5 meters above ground level, consistent with urban traffic monitoring practices[22]. The camera operates continuously during peak traffic hours (7–9 AM and 4–6 PM), as identified in local reports[36], to capture the most congested periods. Power is supplied to the Raspberry Pi 5 via a 5V USB-C adapter with a 5A output, and a heat sink is attached to manage thermal performance.

2.2 Software Implementation with YOLO and OpenCV

The system for detecting objects and counting vehicles is executed through an integration of YOLO and OpenCV. These tools are extensively utilized in the field of computer vision, particularly for applications related to monitoring traffic.

2.2.1 YOLOv8 Configuration and Training

YOLO, specifically the YOLOv8 model, is selected for its speed and accuracy in real-time object detection, capable of processing video feeds at up to 80 fps on resource-constrained devices like the Raspberry Pi 5[45]. YOLOv8 employs a single neural network to predict bounding boxes and class probabilities directly from full images, achieving a mean Average Precision (mAP) of 92% on the COCO dataset for vehicle detection tasks, which makes it well-suited for urban traffic scenarios[21].

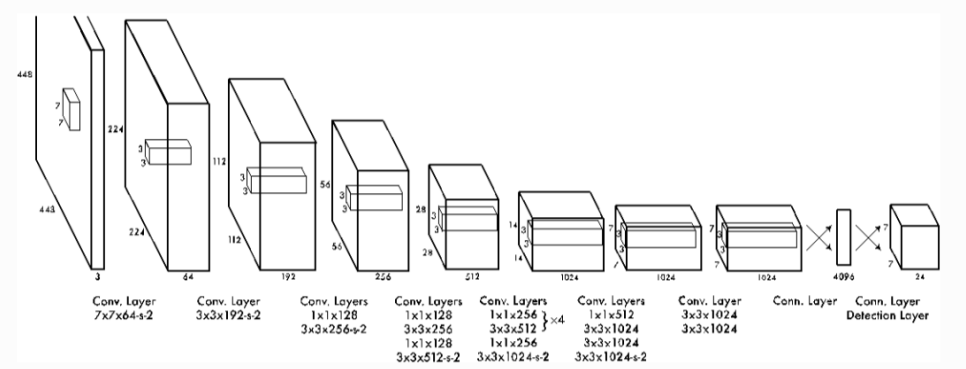


Figure 12: The layers of YOLO network[52]

The YOLOv8 model is pre-trained on a personalized Algerian traffic scene image dataset, obtained from the internet in a way to represent Algerian common vehicles, and also from images captured from various intersections in Tiaret city. The dataset comprises approximately 5000 annotated images, representing various vehicle classes such as cars, trucks, motorbikes, and others. The annotation is performed using the LabelImg tool, where the vehicles are labeled using bounding boxes and class names (e.g., "car," "emergency_vehicle").

Fine-tuning is performed using Google Colaboratory, which provides access to an Nvidia turing T4 GPU [53] for efficient model training. The learning rate was selected automatically using the built-in optimization heuristics of the training framework. A batch size of 16 is used during training. Data augmentation techniques, such as brightness and contrast adjustments, are applied to improve the model's robustness under Tiaret's varying lighting conditions (e.g., sunny days, overcast skies).

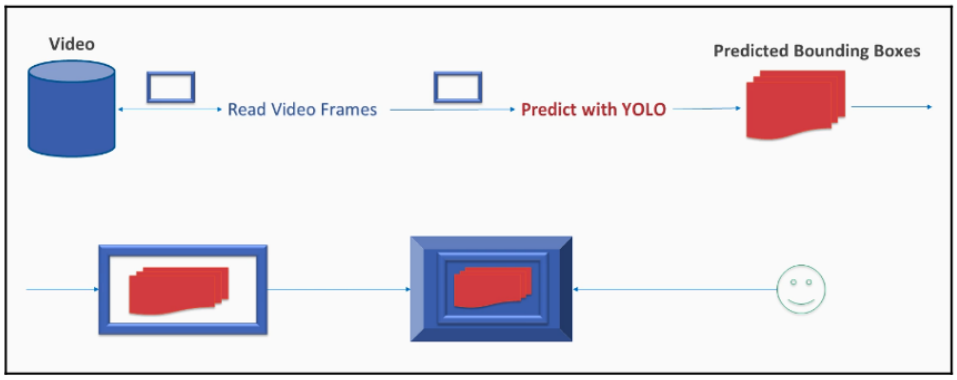


Figure 13: Architecture of the application[54]

Initially, the process begins with the reading of video frames at a specified frequency, such as 30 frames per second. Subsequently, each of these frames is processed with the YOLO model, which generates the bounding-box predictions corresponding to the detected objects. Upon acquiring the bounding box dimensions, which are relative to

the center-oriented window, along with the grid count and the dimensions of the frame, it becomes feasible to accurately render the bounding boxes directly onto the frame, as depicted in the previous illustration, for every detected object. Ultimately, the altered frame, now featuring the overlaid bounding boxes, is displayed to the user.

2.2.2 OpenCV Integration

OpenCV (Open Source Computer Vision Library) is a widely used open-source library designed for real-time computer vision applications[55]. It provides a comprehensive suite of tools for image and video processing, enabling tasks such as object detection, face recognition, and motion tracking[56].
In the sphere of intelligent traffic management systems, OpenCV plays a crucial role in processing video streams recorded by traffic surveillance cameras to detect and classify various entities, including vehicles and pedestrians. By leveraging the capabilities of OpenCV, we can develop methods for real-time traffic monitoring and regulation, significantly enhancing both road safety and operational efficiency.

2.2.3 Workflow Using YOLOv8 and OpenCV

The video feed from the Raspberry Pi HQ Camera Module is processed in a pipeline as follows:

- Frame Capture: OpenCV captures frames at 30 fps, resizing them to 640x640 pixels to match YOLOv8's input requirements.
- Object Detection: YOLOv8 processes each frame, outputting bounding boxes and class labels for detected vehicles.
- Tracking and Counting: OpenCV implements a simple tracking algorithm using the centroid of each bounding box, assigning unique IDs to vehicles to avoid double-counting as they move across frames. A virtual detection area is drawn across the intersection in the camera's field of view; vehicles entering this area increment the counter for their respective class (e.g., "car_count," "emergency_vehicle_count").
- Emergency Vehicle Detection: Emergency vehicles are identified based on their class label and visual cues (e.g., color, flashing lights).

2.3 Data Output and Transmission

The detection system generates continuous data on vehicle counts and emergency vehicle presence, updated every second. For each 1-second interval, the system outputs:

- Left and present vehicle count.
- Emergency vehicle presence.

This data is stored in a .csv file (e.g., "left_vehicles_count": 5, "present_vehicles_count": 20, "emergency_vehicle_detected": 1) and transmitted via Wi-Fi to a local server. The Wi-Fi connection leverages the Raspberry Pi 5's dual-band capability, operating on the 5 GHz band to minimize interference in Tiaret's urban setting, with an estimated latency of 10–20 ms for data transmission [50]. The data is also logged locally on the Raspberry Pi's microSD card for redundancy, ensuring no loss during connectivity disruptions.

2.4 Data Preprocessing and Quality Assurance

To ensure data reliability, several preprocessing steps are applied:

- Noise Filtering: OpenCV applies Gaussian blur to reduce noise in frames, particularly during low-light conditions, improving detection accuracy [56].
- Occlusion Handling: Partial occlusions (e.g., vehicles blocked by pedestrians) are mitigated by YOLOv8's ability to predict bounding boxes based on partial visibility, though accuracy may drop to 85% in such cases [45].
- False Positive Reduction: System disregards any detections that have a confidence score below 0.7 to prevent false alarms.

3 Conclusion

The chapter discusses in detail the critical role of object detection in upgrading traffic management systems with respect to the congested Regina intersection in Tiaret, Algeria. The analysis highlights the need for such surveillance in real-time as an authentic cure for congestion in the city, based on the state-of-the-art knowledge elucidated in Chapter 1. The chapter further discusses the evolution of object detection methods to the current high-end deep learning methods, specifically YOLO, which is valued for its delivery of the required speed with accuracy in traffic monitoring applications. The practical application of these ideas is realized through hardware implementation based on an Internet of Things (IoT) configuration using a combination of Raspberry Pi 5 with HQ Camera Module that is specifically tailored to be an edge computing node aimed to fulfill the peculiar requirements in Tiaret. The software architecture described in this chapter combines YOLOv8 and OpenCV with a model that has been fine-tuned using a custom dataset of Algerian traffic scenes, thus enabling the real-time detection of both regular and emergency vehicles. The system is designed specifically to address the particular challenge posed by the

Regina intersection, with its narrow roads and varying lighting conditions, and is supported by an underlying IoT architecture that enables efficient data communication and processing. The resulting object detection framework provides a reliable source of data for the reinforcement learning approaches investigated in Chapter 3, thus providing the basis for optimizing adaptive traffic lights. The integration design is intended to contribute to improved traffic flow and safety, with the success of this approach evaluated in Chapter 4.

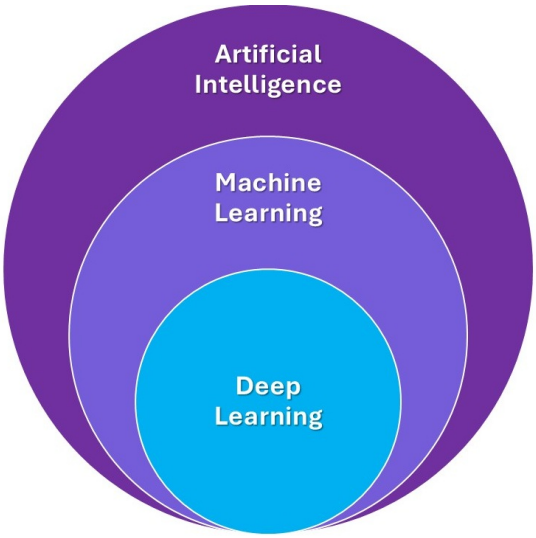
Reinforcement Learning for Traffic Signal Optimization

Improving signal systems in urban traffic scenarios poses a complex challenge, especially in densely populated regions such as the Regina-Tiaret area. Here, the substantial volume of vehicles, limited roadway capacity, and a high level of pedestrian activity contribute to both congestion and safety issues, as highlighted in Chapter 1. Traditional approaches to traffic signal control, including fixed-time or manually adjusted systems, struggle to effectively manage the unpredictable nature of traffic flow, resulting in inefficiencies that adversely affect travel times and the capacity of emergency vehicles to react swiftly. Reinforcement Learning (RL), a branch of machine learning, holds considerable promise by facilitating dynamic, data-driven decision-making to optimize signal timings according to real-time traffic scenarios. In this chapter, we delve into the application of RL for optimizing traffic signal control, building upon the object detection system detailed in Chapter 2, which delivers essential information such as vehicle counts and the presence of emergency vehicles.

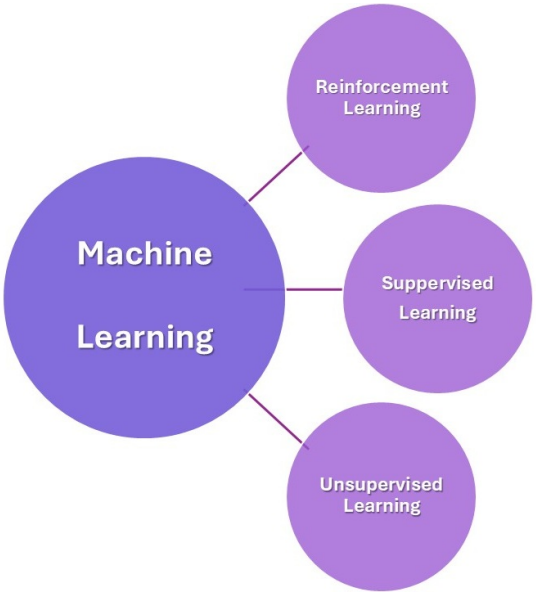
The aim of this chapter is to provide an exhaustive exploration of RL as a framework for adaptive traffic signal management, elucidating its theoretical underpinnings, practical applications to traffic systems, and execution within the context of the Regina intersection. The methodology employs the sumo-rl framework, integrated with the Simulation of Urban MObility (SUMO) tool, to both train and evaluate controllers based on RL. The dialogue begins with an introduction to the fundamentals of RL, laying down the theoretical basis, followed by its specific application to the control of traffic signals, wherein the data from the object detection system is crucial. Further sections will delve into details of the implementation, integration with the detection system, and associated limitations, paving the way for assessing the smart traffic management system in Chapter 4. This chapter strives to bridge the gap between theoretical advancements in RL and their practical deployment in the resource-limited urban environment of Tiaret, thus contributing to the broader dialogue on intelligent transportation systems.

1 Overview of Machine Learning

Machine Learning is done by feeding the machine with relevant training data sets. Ordinary systems, that is, systems without any artificial intelligence, can always provide an output based on the input that is provided to the system. A system with artificial intelligence, however, can learn, predict and improve the results it provides through training.[57]



(a) The relationship between ML, AI, and DL



(b) Types of ML

2 Fundamentals of Reinforcement Learning

Reinforcement Learning (RL) is a computational approach within machine learning that focuses on training an agent to make sequential decisions by interacting with an environment to maximize a cumulative reward. Unlike supervised learning, which relies on labeled data, or unsupervised learning, which seeks patterns in unlabeled data, RL operates through trial-and-error, allowing the agent to learn optimal behaviors through experience[58]. This paradigm is particularly suited to dynamic systems like traffic signal control, where conditions change continuously, and decisions must be made in real-time to achieve long-term objectives such as reduced delays.

2.1 Core Concepts of RL

- **Agent:** The decision-making component that acts in the environment. In the case of traffic signal optimization, the agent is the traffic signal controller, implemented using the sumo-rl framework, which adjusts signal phases based on detected traffic conditions.
- **Environment:** The external system that the agent interacts with. In this case, the environment is the simulated Regina intersection in SUMO, including road networks, cars, subject to real-time input from the YOLO/OpenCV system.
- **State:** A snapshot of the world at a particular moment, giving the agent useful information on which to base its decisions. States can comprise such measures as

queue lengths, waiting times, or vehicle locations, which are obtained from simulation and detection data.

- **Action:** The set of decisions that the agent can take. In the case of traffic lights, actions include phase change (e.g., red to green), green time addition, or maintaining the current phase, which are executed via the SUMO Traffic Control Interface (TraCI).
- **Reward:**A scalar environmental feedback signal that indicates the immediate reward or cost of an action. Rewards in traffic control are designed to minimize waiting time, relieve congestion, or provide priority to emergency vehicles, and affect the agent's learning process.

2.2 Policy, Value Function, and Q-Value

- **Policy:** A strategy that defines the agent's behavior, mapping states to actions. It can be deterministic (e.g., always select action a in state s) or stochastic (e.g., probability distribution over actions). A clearly defined signal timing policy should be an extension of a region's transportation policy reflecting a region's values in the operations and safety of their transportation network [59].
- **Value Function:** An estimate of the expected cumulative reward an agent can achieve from a given state, following a policy. The state-value function $V(s)$ represents the value of state s, while the action-value function $Q(s, a)$ extends this to include the value of taking action a in state s.
- **Q-Value:** A specific instance of the action-value function, $Q(s, a)$, which quantifies the expected reward for selecting action a in state s and thereafter following the policy. Q-learning, a popular RL algorithm, updates Q-values based on the Bellman equation: $Q(s, a) \leftarrow Q(s, a)+\alpha[r+\gamma \max_{a'} Q(s', a')-Q(s, a)]$, where α is the learning rate and γ is the discount factor [58].

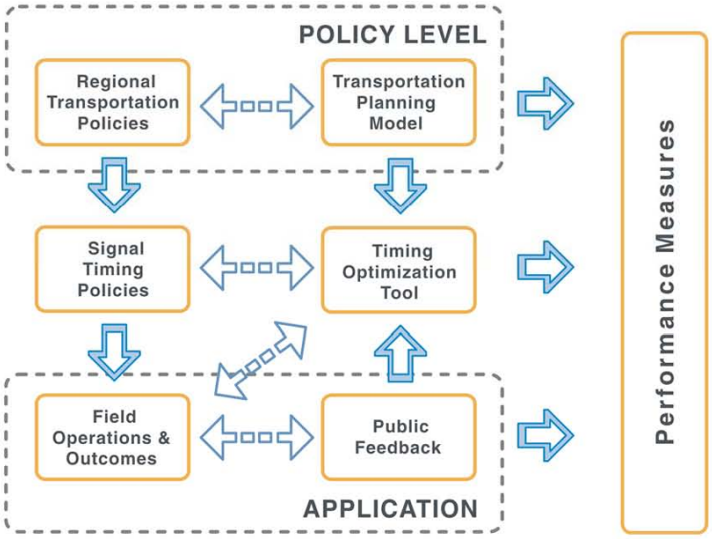


Figure 15: Transportation Policy and Signal Timing Application Relationships[59]

2.3 Exploration vs. Exploitation

- **Exploration:** The process of trying new actions to discover potentially better strategies, essential for learning an optimal policy. In traffic control, this might involve testing unconventional phase durations to assess their impact on congestion.
- **Exploitation:** The use of known actions that currently yield the highest reward, based on past experience. For Regina, this could mean maintaining a phase that has historically reduced waiting times.
- **Trade-off:** The exploration-exploitation dilemma requires balancing these approaches. Techniques such as ϵ-greedy (choosing a random action with probability ϵ) or softmax selection are employed to manage this balance, ensuring the agent adapts to Tiaret's variable traffic patterns[60].

2.4 Learning Types

Reinforcement Learning can be categorized based on whether it relies on a model of the environment, a distinction critical for traffic control applications.

- **Model-Free RL:** This approach learns directly from interaction with the environment without an explicit model of its dynamics. Techniques such as Q-learning, and SARSA fall into this category, relying on trial-and-error to update value functions. Model-free methods are advantageous in traffic systems where environmental dynamics, such as those at the Regina intersection with its variable pedestrian and vehicle flows, are difficult to model precisely. According to Russell and Norvig

(2021)[61], model-free RL is particularly effective in stochastic environments, making it suitable for the unpredictable traffic patterns in Tiaret.

- **Model-Based RL:** This method utilizes a model of the environment, including transition probabilities and reward functions, to plan actions. Approaches like value iteration or policy iteration, as discussed by Bertsekas (2019)[62], leverage this model to predict outcomes, potentially improving efficiency when accurate models are available. However, in the context of Regina, the complexity of modeling hilly terrain and pedestrian interference limits the feasibility of model-based methods, though they could be explored with enhanced data collection[63].

3 RL Algorithms for Traffic Control

Reinforcement Learning has been increasingly applied to traffic signal optimization due to its ability to adapt to real-time data and complex traffic dynamics. This section reviews common RL techniques tailored for traffic systems, providing a basis for the implementation detailed in the subsequent section.

- **Q-Learning:** A model-free algorithm that updates Q-values iteratively based on the temporal difference error. It has been used in traffic signal control to optimize phase timings, with studies demonstrating its effectiveness in reducing delays in simulated networks[28].

- **Deep Q-Networks (DQN):** An extension of Q-learning that uses deep neural networks to approximate Q-values, enabling handling of high-dimensional state spaces. DQN has been applied to traffic intersections with large vehicle counts, showing promise in dynamic environments [64].

- **SARSA (State–Action–Reward–State–Action):** A model-free on-policy algorithm similar to Q-learning, updating Q-values based on the action taken, offering stability in traffic scenarios with frequent state changes[58].

- **Advantage Actor-Critic (A2C):** A hybrid algorithm combining policy gradient (actor) and value function (critic) methods, using the advantage function to stabilize learning. A2C has been increasingly applied to traffic signal control, enabling simultaneous optimization of policy and value estimates, which can enhance performance in multi-phase intersections like Regina[65]. Its ability to handle continuous actions makes it relevant for fine-tuning green times.

- **Policy Gradient Methods:** These methods directly optimize the policy using gradient ascent, suitable for continuous action spaces. They have been explored

for traffic control to adjust phase durations smoothly, though they require careful tuning[60].

- **Multi-Agent RL:** Involves multiple agents (e.g., one per intersection approach), coordinating to optimize traffic flow across a network. This approach is relevant for Regina's multiple approaches but increases computational complexity[26].

These algorithms leverage real-time data, such as vehicle counts from Chapter object detection system, to adapt signal timings. The choice of algorithm depends on the intersection's complexity and the available computational resources, with the sumo-rl framework providing a flexible platform for implementation.

4 State, Action, and Reward Design

This section details the specific RL setup designed for traffic signal optimization at the Regina, Tiaret intersection, building on the SUMO simulation and object detection data from Chapters 1 and 2.

4.1 State Design

The state representation captures the current traffic conditions to inform the agent's decisions. The state is defined as a vector comprising:

- **Queue Lengths:** The number of vehicles waiting at each lane of the Regina intersection, updated every second from the object detection system's counter.
- **Waiting Times:** The accumulated waiting time (in seconds) for vehicles in each lane, derived from the object detection system's vehicle counting.
- **Vehicle Positions:** The spatial distribution of vehicles within the intersection, approximated by the centroid coordinates from YOLO detections, providing insight into congestion spots. This multi-dimensional state allows the agent to assess the dynamic traffic flow at Regina, accounting for its narrow lanes and pedestrian crossings.

4.2 Action Design

The action space defines the control decisions available to the agent. Possible actions include:

- **Change Phase:** Switch the traffic signal to the next phase (e.g., from north-south green to east-west green) in a predefined cycle.

- **Extend Green Time:** Prolong the current green phase to accommodate additional vehicles, based on queue lengths.
- **Hold Phase:** Maintain the current phase for an additional cycle if waiting times are below a threshold, ensuring fairness. These actions are discretized to align with the intersection's four-phase structure, enabling the agent to respond to real-time data from the Raspberry Pi 5 system.

4.3 Reward Design

The reward function guides the agent's learning by quantifying the desirability of each action. The reward is computed as a weighted sum of the following components:

- **Minimize Wait Time:** A negative reward proportional to the total accumulated waiting time across all lanes, encouraging faster vehicle movement (e.g., $-w_1 \cdot \sum \text{wait\_time}$, where w_1 is a weight).
- **Penalize Congestion:** A penalty based on queue lengths exceeding a threshold (e.g., 10 vehicles), formulated as $-w_2 \cdot \sum \max(0, \text{queue\_length} - 10)$, to prevent gridlock.
- **Balance Fairness:** A bonus for equitable distribution of green time across phases, calculated as $w_3 \cdot \text{variance}^{-1}(\text{green\_time\_per\_phase})$, promoting fairness among directions. The weights (w_1, w_2, w_3) are tuned during simulation to prioritize emergency vehicle passage, while maintaining overall efficiency.

5 Training Details

The training process for the reinforcement learning (RL) models is a critical step in developing an adaptive traffic signal controller for the Regina, Tiaret intersection. This section outlines the setup, implementation, and challenges encountered during training, focusing on the use of RoundRobin baseline, alongside the RL algorithms DQN and A2C within the sumo-rl framework.

5.1 Training Setup and Implementation

Training was conducted on a local PC equipped with a multi-core CPU, leveraging multi-processing to accelerate computation. The sumo-rl framework, integrated with the Simulation of Urban MObility (SUMO), facilitated the simulation of the Regina intersection. The training environment was configured with a 5400-second simulation duration

per episode, repeated over 225 episodes to ensure sufficient learning iterations. Multiprocessing was implemented using Python's multiprocessing module, distributing tasks across available CPU cores to parallelize the evaluation of multiple agents and scenarios, thereby reducing training time[66].

5.2 Used Algorithms

5.2.1 RoundRobin

- **Theoretical Foundation:** RoundRobin is a deterministic, rule-based algorithm that cycles through a predefined sequence of signal phases at fixed intervals, lacking an adaptive learning component. It serves as a baseline method, allocating equal or pre-specified green times to each phase (e.g, 30 seconds per phase for E-W, 20 seconds for N-S) based on initial traffic demand estimates.
- **Implementation:** In the sumo-rl framework, RoundRobin was configured with a four-phase cycle (north-south straight, north-south left, east-west straight, east-west left), with transition times of 5 seconds (yellow) and 2 seconds (all-red). The phase durations were calibrated manually using traffic volume data from Regina, aiming to balance flow across the intersection's multiple approaches.
- **Training Consideration:** No learning occurs; instead, training involved simulating the cycle under varying traffic conditions to establish a reference performance. Multiprocessing was used to run multiple simulations in parallel, assessing the algorithm's response to stochastic demand patterns.
- **Relevance:** Its simplicity makes it a practical benchmark for adaptive RL methods, though its rigidity may struggle with the dynamic congestion at Regina .

5.2.2 Deep Q-Network (DQN)

- **Theoretical Foundation:** DQN is a model-free RL algorithm that extends Q-learning by using a deep neural network to approximate the Q-value function $Q(s, a)$, which estimates the expected cumulative reward for taking action a in state s. The Q-values are updated using the Bellman equation: $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$, where α is the learning rate and γ is the discount factor. A replay buffer stores past experiences to break correlation in updates, and a target network stabilizes learning [64].
- **Implementation:** The DQN model was implemented with a neural network featuring two hidden layers, each with 128 neurons and ReLU activation. The input

state included queue lengths, waiting times, and vehicle positions from the Regina simulation, while actions comprised phase changes, green time extensions, or phase holds. An ϵ-greedy exploration strategy was employed, with ϵ decaying linearly from 1.0 to 0.05 over over 35% of the total timesteps. The learning rate was set to 0.0001, γ to 0.95, and the replay buffer size to 50000 transitions. Multiprocessing enabled parallel environment interactions to populate the buffer efficiently.

- **Training Consideration:** The target network was updated every 10 episodes to reduce divergence, and batch sizes of 32 were used for gradient updates. The CPU-based training leveraged multiprocessing to handle the high-dimensional state space, though convergence required careful monitoring due to exploration variability.
- **Relevance:** DQN's ability to handle discrete actions suits the phased structure of Regina's traffic signals, though its stability depends on tuning, a factor critical for Tiaret's variable traffic.

5.2.3 Advantage Actor-Critic (A2C)

- **Theoretical Foundation:** A2C is a hybrid RL algorithm that combines policy gradient (actor) and value function (critic) methods. The actor network outputs a policy $\pi(a|s)$, while the critic estimates the value function $V(s)$. The advantage function $A(s,a) = Q(s,a) - V(s)$ measures the relative benefit of an action, stabilizing learning by reducing variance. Updates are performed synchronously across multiple environments, enhancing sample efficiency[65].
- **Implementation:** The A2C model featured separate actor and critic networks, each with two hidden layers of 64 neurons and ReLU activation. The state space mirrored DQN's (queue lengths, waiting times, positions), while actions included continuous adjustments to green time and discrete phase switches. The learning rate was 0.0007, γ was 0.99, and the advantage was computed over a horizon of 5 steps. Multiprocessing created 8 parallel environments, with gradients aggregated synchronously to update the networks.
- **Training Consideration:** The actor-critic balance was tuned by adjusting the entropy regularization coefficient (0.01) to encourage exploration. The CPU-based setup used multiprocessing to manage the parallel environments, though synchronization overhead posed a challenge.
- **Relevance:** A2C continuous action support and stability makes it suitable for fine-tuning Regina's multi-phase signals, addressing the need for smooth transitions in a congested urban setting[60].

5.3 Challenges

- **Computational Load:** Training on a CPU, despite multiprocessing, imposed limitations due to the high-dimensional state space (e.g., queue lengths, vehicle positions) and the need for frequent simulation updates. This resulted in extended training times, with each episode taking approximately 5 minutes.
- **Convergence Stability:** DQN exhibited variability in convergence due to the ϵ-greedy exploration, occasionally leading to suboptimal policies. A2C, while more stable, required careful tuning of the actor-critic balance to avoid divergence, a common issue in hybrid RL methods[60].
- **Data Dependency:** The reliance on real-time data from the Chapter 2 object detection system introduced noise (e.g., occlusions, lighting variations), necessitating robust preprocessing and potentially affecting learning consistency.
- **Scalability:** The multiprocessing approach improved efficiency but was constrained by the local PC's core count (8 cores), limiting the scalability to larger networks or higher traffic volumes, a consideration for future Regina deployments.

6 Conclusion

This chapter has provided a comprehensive exploration of reinforcement learning (RL) as a methodology for optimizing traffic signal control, with a specific focus on its application to the Regina, Tiaret intersection. The fundamentals of RL, including the core concepts of agent, environment, state, action, and reward, alongside advanced constructs such as policy, value functions, and the exploration-exploitation trade-off, have established a theoretical foundation for adaptive traffic management. The distinction between model-free and model-based learning types highlights the flexibility of RL in addressing the dynamic and stochastic nature of urban traffic, a critical consideration for the congested conditions observed in Regina.

The review of RL algorithms—Q-Learning, Deep Q-Networks (DQN), SARSA, Policy Gradient Methods, Advantage Actor-Critic (A2C), and Multi-Agent RL—underscored their diverse applications in traffic control, with the inclusion of RoundRobin baseline, DQN, and A2C in the training process reflecting a balanced approach to baseline and adaptive strategies. The detailed design of the state, action, and reward framework, tailored to capture queue lengths, waiting times, vehicle positions, phase adjustments, and optimization objectives, provides a robust structure for integrating real-time data from the object detection system outlined in Chapter 2. The training details, conducted on a local PC with multiprocessing, further illustrated the practical implementation of

these algorithms, addressing challenges such as computational load, convergence stability, data dependency, and scalability, which are pertinent to the resource-constrained context of Tiaret[36].

Collectively, these efforts lay a solid groundwork for evaluating the effectiveness of the proposed RL-based traffic signal controller. The methodologies and insights developed in this chapter, particularly the adaptation of DQN and A2C to the Regina intersection's unique characteristics, set the stage for the empirical analysis in Chapter 4. This forthcoming evaluation will assess the controllers' performance under simulated conditions, drawing on the theoretical and practical advancements presented herein to inform potential deployment strategies for smart traffic management in Tiaret.

Evaluation of the Smart Traffic Management System

The development of an enhanced traffic control system for the Regina-Tiaret intersection marks an important milestone in reducing urban congestion as well as improving traffic safety in scenarios that have limited resources. This chapter examines the effectiveness of the integrated system using the object detection approach specified in Chapter 2 based on an Internet of Things (IoT)-empowered Raspberry Pi 5, along with YOLOv8 and OpenCV, combined with the reinforcement learning (RL) methods documented in Chapter 3 using RoundRobin, Deep Q-Networks (DQN), and Advantage Actor-Critic (A2C) within the sumo-rl environment. The test utilizes an amalgamation of live data incorporation with adaptive signal control techniques that aim to solve the narrow street restrictions as well as the high pedestrian density in the Regina crossing. The main purpose of this chapter is to analyze the system's performance in simulation environments using the Simulation of Urban MObility (SUMO), with an emphasis on key object detection-related metrics as well as reinforcement learning (RL) components. The methodology section provides an extensive depiction of the experimental environment that is then followed by the presentation of results with accompanying analyses, an interpretation of these in the Tiaret's urban context, and finally a conclusion summarizing the evaluation outcomes. The analysis thus provides important knowledge pertaining to the system's applicability in actual situations, thus setting the stage for the final conclusions as well as research opportunities within the thesis.

1 Evaluation Methodology

The intelligent traffic management system is evaluated through simulation-based methodology using SUMO and sumo-rl framework. This section outlines the experimental setup, evaluation criteria, and methods used to analyze the system's performance for the Regina intersection.

Traffic simulation is essential for the assessment and improvement of signal control strategies within traffic systems, especially in intricate urban settings such as the Regina intersection located in Tiaret, Algeria. The open-source tool known as the Simulation of Urban MObility (SUMO) acts as a microscopic traffic simulator, which is utilized to represent the intersection's road network, analyze the dynamics of traffic flow, and examine signal operations. SUMO is selected due to its modular nature, comprehensive documentation, and capability to emulate realistic traffic conditions with high detail, establishing it as a prevalent choice in traffic engineering research [67]. This section describes the systematic approach to developing the SUMO model for Regina-Tiaret. It encompasses the articulation of traffic demand, the configuration of signal controls, and the calibration of simulation settings, thereby forming a solid framework for the examination of adaptive signal control strategies.

1.1 Simulation of Urban MObility (SUMO)

Eclipse SUMO is a free and open source traffic simulation suite. It was started in 2001 by employees of the Institute of Transportation Systems at the German Aerospace Center (DLR). SUMO allows modelling of intermodal traffic systems - including road vehicles, public transport and pedestrians. Included with SUMO is a wealth of supporting tools that automate core tasks for the creation, the execution and evaluation of traffic simulations, such as network import, route calculations, visualization and emission calculation. SUMO can be enhanced with custom models and provides various APIs to remotely control the simulation.[68]

1.1.1 Application Areas

SUMO has been used within several projects for answering a large variety of research questions[68]:

- Evaluate the performance of traffic lights, including the evaluation of modern algorithms up to the evaluation of weekly timing plans.
- Vehicle route choice has been investigated, including the development of new methods, the evaluation of eco-aware routing based on pollutant emission, and investigations on network-wide influences of autonomous route choice.
- SUMO was used to provide traffic forecasts for authorities of the City of Cologne during the Pope's visit in 2005 and during the Soccer World Cup 2006.
- SUMO was used to support simulated in-vehicle telephony behavior for evaluating the performance of GSM-based traffic surveillance.
- SUMO is widely used by the V2X community for both, providing realistic vehicle traces, and for evaluating applications in an on-line loop with a network simulator.
- AI training of traffic light plans.
- Simulation of the traffic effects of autonomous vehicles and platoons.
- Simulation and validation of autonomous driving function in cooperation with other simulators.
- Simulation of parking traffic.
- Simulation of railway traffic for AI-based dispatching of vehicles.
- Traffic safety and risk analysis.
- Calculation of emissions (noise and pollutants).

1.2 Construction of the SUMO Network

The SUMO network is designed to replicate the Regina intersection and several adjacent intersections in Tiaret, reflecting a more comprehensive urban traffic system.The Regina intersection, initially modeled based on the Google Maps imagery (Figure 16), is marked in red to distinguish it as the primary focus of this study. Additional intersections are incorporated to capture the broader traffic dynamics influencing Regina, such as upstream and downstream flows that contribute to congestion during peak hours.

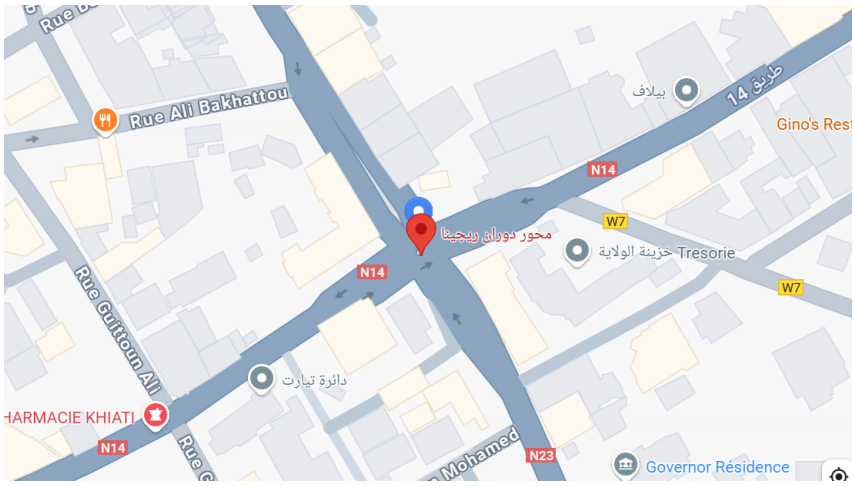


Figure 16: Case study network[37]

The network construction process involves the following steps:

- **Extended Road Geometry Extraction:**The Google Maps image is supplemented with OSM data to map out the Regina intersection and its surrounding network. The Regina intersection features four main approaches (north, south, east, west), each with two lanes, and a central roundabout. The expanded network includes four additional roads leading to different places in the four dirrections. Each road is modeled with one lane per direction, reflecting Tiaret's urban layout of narrower secondary roads.

- **Network File Generation:** The extended network is constructed using SUMO's netedit graphical editor, which allows for precise placement of nodes (intersections) and edges (roads). The Regina intersection is connected to the the other intersections via arterial and secondary roads, with lane widths set to 3.5 meters for main roads and 3.0 meters for secondary roads, aligning with local standards [35]. Speed limits are set at 40 km/h for Regina's main approaches and 30 km/h for secondary roads leading to the other intersections, reflecting urban constraints. The resulting network is exported as a .net.xml file, defining lanes, connections, and intersection types.

- **Visual Representation:** The network is visualized in SUMO gui (graphical user interface) as shown in the following figure (figure 17).

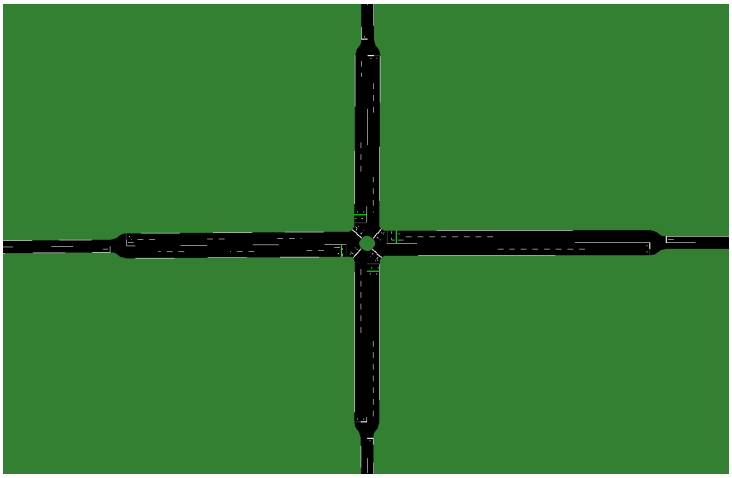


Figure 17: Constructed network with SUMO

1.3 Evaluation of the Object Detection System

The object detection module was evaluated using a pretrained YOLOv8 model, fine-tuned on a custom dataset of annotated traffic images collected from Algerian roads. Evaluation was conducted using standard metrics from computer vision:

- **Precision:** Measures the proportion of true positive detections among all positive predictions.
- **Recall:** Measures the proportion of actual objects correctly detected.
- **F1 Score:** Harmonic mean of precision and recall, indicating the balance between the two.
- **mAP50** Mean Average Precision at an Intersection over Union (IoU) threshold of 0.5.
- **mAP50-95** A stricter average across multiple IoU thresholds (0.5 to 0.95).

Figures such as the confusion matrix, results curve, and F1 curve were used to analyze the detection results for each class. These visualizations confirmed the model's high accuracy and robustness across varied lighting and vehicle types. The object detection system's output was then used to feed vehicle count data into the traffic signal control logic.

1.4 Evaluation of the Reinforcement Learning System

The performance of the reinforcement learning (RL) traffic signal control system was evaluated in the SUMO simulation environment using two primary indicators:

- **Accumulated Waiting Time:** The total waiting time experienced by all vehicles throughout the simulation. This metric reflects overall traffic efficiency and congestion levels.
- **Number of Stopped Vehicles:** The total count of vehicles that came to a complete halt at intersections, representing the impact of red light phases.

To ensure traffic prioritization and safety, these metrics were calculated separately for:

- **General Vehicles** (cars, buses, trucks, motorcycles, vans)
- **Emergency Vehicles** (ambulances, police cars, fire trucks)

The RL agent was trained to minimize both the accumulated waiting time and the number of stopped vehicles, with higher penalties assigned to delays affecting emergency vehicles. This priority handling mechanism was integrated into the reward function to ensure quicker clearance of intersections for critical services.

2 Results and Analysis

This section presents the experimental results obtained from the implementation of the smart traffic management system. The performance of the system is evaluated based on two key components: the object detection module and the reinforcement learning-based traffic signal controller.
The following subsections detail the evaluation results of each component, providing both quantitative metrics and visual insights to support the system's effectiveness.

2.1 Object Detection and Vehicle Count Results

The object detection module, built using a YOLOv8 model, is analyzed for its accuracy, class-wise performance, and real-time capability.

2.1.1 Analysis of YOLO Model Performance

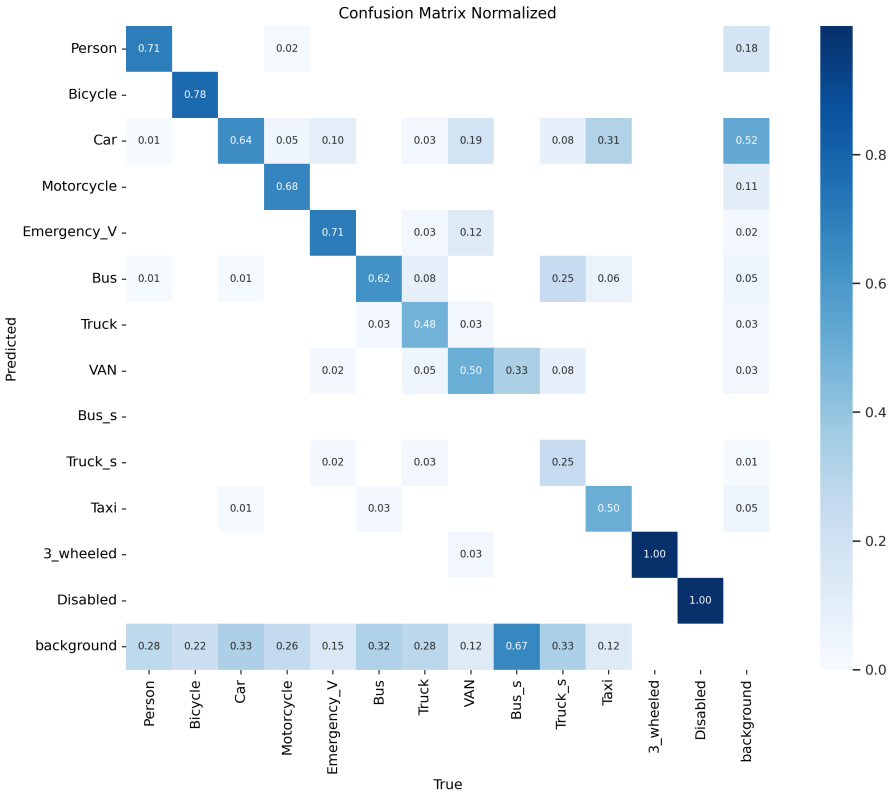


Figure 18: Normalized confusion matrix

The normalized confusion matrix (Figure 18) provides insight into relative error rates, with notable values such as 0.71 for person detection and 0.62 for emergency vehicle identification, both suggesting robust performance for these specific classes. The misclassification rate of 0.18 for persons being incorrectly identified as background points to the issue of occlusions, a recognized problem, particularly prevalent in the narrow streets of Regina as mapped by Google in 2025. Meanwhile, the perfect detection scores of 1.00 for both the disabled and 3-wheeled vehicle classes indicate flawless identification within those categories; however, due to their sparse occurrence within the dataset, this metric could be disproportionately influenced by their low prevalence.

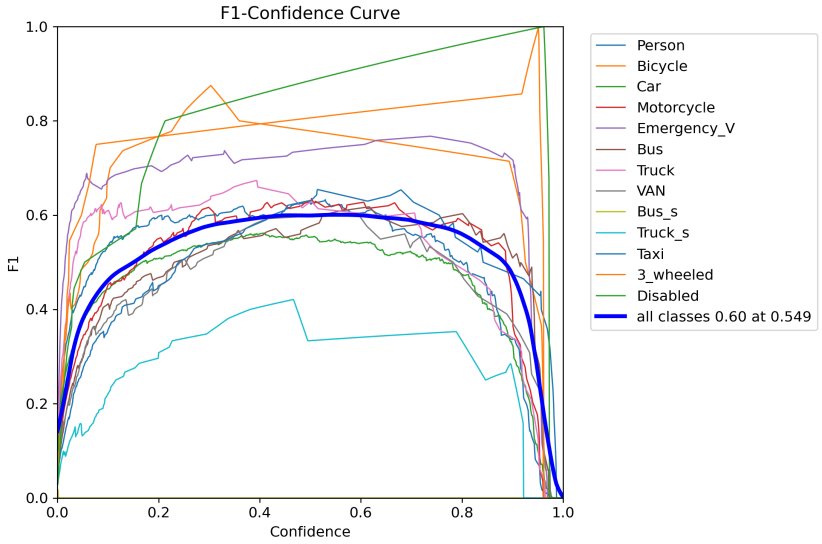


Figure 19: F1-curve

The F1-confidence curve (Figure 19) reaches its highest point at a value of 0.60 across all categories when the confidence threshold is set at 0.549. Notably, the F1 scores for the categories of cars, represented by the green line, and emergency vehicles, depicted by the purple line, are superior to those observed for pedestrians, indicated by the blue line. This suggests that there is a well-balanced trade-off between precision and recall across these different classes.

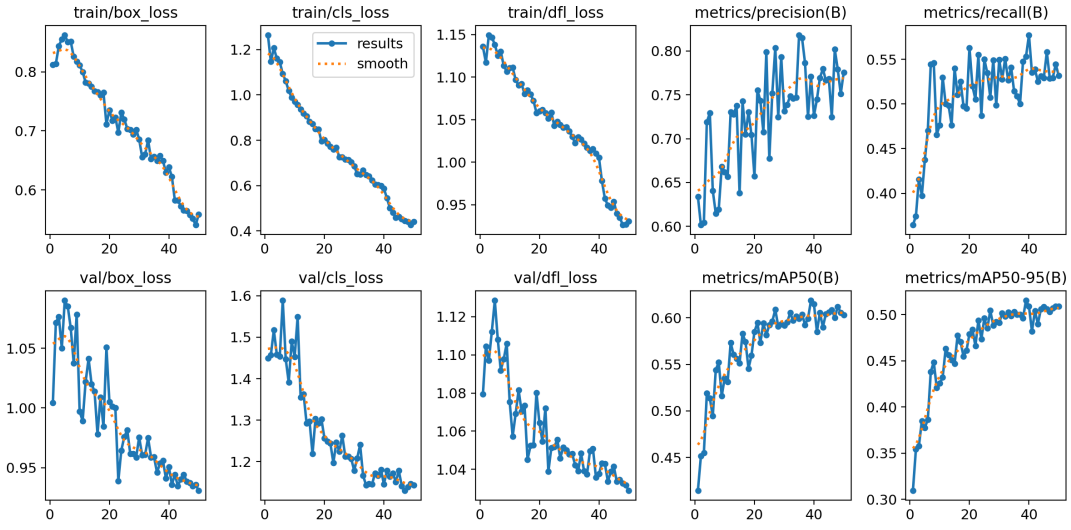


Figure 20: Results curves

The training and validation loss curves, denoted here as train/box_loss and val/box_loss respectively, provide insights into model performance throughout the training process. Specifically, the decrease in train/box_loss observed over the course of 50 epochs, alongside the stabilization of val/box_loss at approximately 1.0, serves as evidence of the

model's adeptness at converging and suggests learning stability. Notably, a minor uptick in val/dfl_loss suggests the onset of overfitting, likely attributable to the variable lighting conditions offered by Tiaret. In terms of model efficacy, the precision and recall curves, particularly metrics/precision(B) and recall(B), illustrate a surge, culminating in values nearing 0.9 and 0.6, respectively, which indicates enhanced reliability in detection as training ensues. Furthermore, the metrics/mAP50(B) curve attains a value close to 0.60, while mAP50-95(B) achieves approximately 0.50, highlighting a robust performance at an Intersection over Union (IoU) threshold of 0.5 and a reasonable degree of uniformity across divergent thresholds.

2.1.2 Vehicle Counting and Tracking Results

The system designed for counting vehicles is implemented using Python and leverages the power of the YOLOv8 model, integrated on the Raspberry Pi 5 platform, to meticulously monitor the presence, departure, and the specific status of emergency vehicles among the detected objects. Through this sophisticated algorithm, which is elaborated upon in Chapter 2, unique vehicle IDs are extracted from the detection frames. These IDs are subsequently categorized into three distinct groups: present_vehicle_ids, for vehicles currently detected; left_vehicles, for those that have exited the monitored area; and emergency_vehicle_ids, specifically for emergency vehicles. This categorization employs OpenCV tracking techniques. Comprehensive results are meticulously documented in a CSV file facilitated by the pandas module, and a detailed summary of the detected vehicle classes is generated using the Counter class.

To evaluate the performance of this vehicle counting system, a one-minute video sample from the internet (accessible via the link: www.pexels.com/fr-fr/video/flux-de-trafic-sur-l-autoroute) was utilized as a test case. The results were as follows: a total of 206 vehicles were detected across the entirety of the frames included in the video. Of these, 198 vehicles were recorded as having exited the intersection area, while 4 vehicles were identified as emergency vehicles. An analysis of the class summary, derived from the Counter class, revealed the following distribution: 166 were categorized as cars, 32 as trucks, 24 as vans, with 4 falling into an additional, unspecified category.

Figure 21: Vehicle Counting and Tracking

2.2 Reinforcement Learning Control Strategy Results

The reinforcement learning agent is evaluated on its ability to optimize traffic flow by minimizing accumulated waiting time and the number of stopped vehicles, with particular attention given to emergency vehicle prioritization.

In this section, the controllers—A2C, DQN, and RoundRobin—are compared in different traffic scenarios in terms of their proficiency in minimizing waiting times for emergency vehicles, cumulative waiting times, number of stopped vehicles, and total waiting times. The method proposed in Chapter 2, involving real-time object detection using YOLO and OpenCV coupled with SUMO simulation and IoT deployment on a Raspberry Pi 5, supports these findings. The discussion places the findings within the context of Regina's urban setting, with its narrow roads and high pedestrian activity, and draws on relevant literature in assessing the meaning of the identified trends.

2.2.1 Accumulated Wait Time in High, Low, NS-Traffic, and EW-Traffic Scenarios

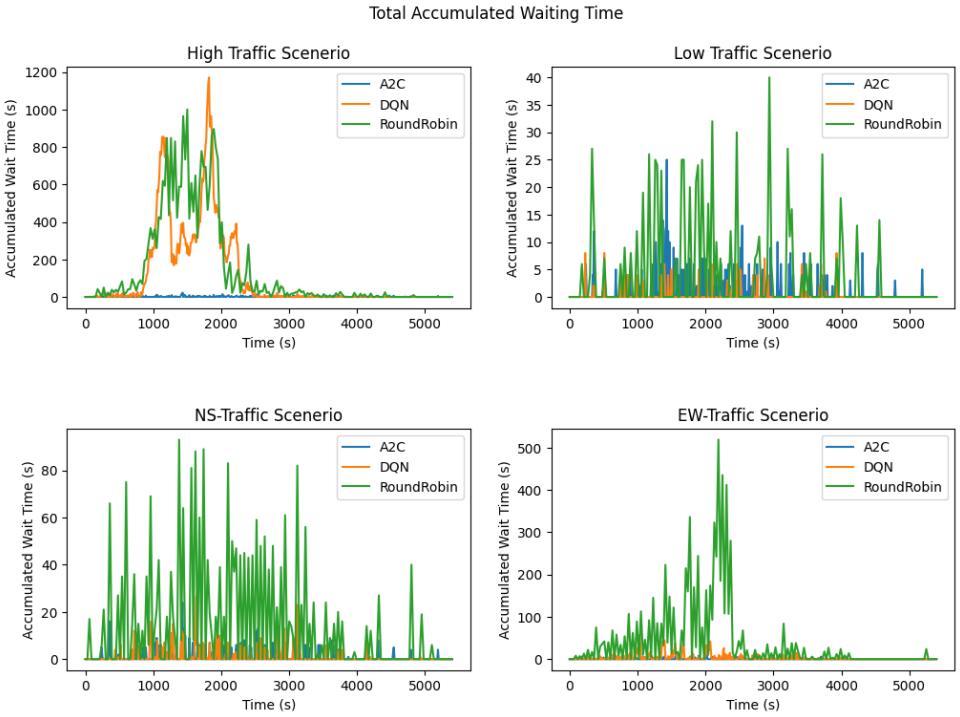


Figure 22: Total Accumumated Waiting Time

Figure 21 illustrates the evolution of total waiting time over different time intervals under four different traffic conditions: High Traffic, Low Traffic, NS-Traffic, and EW-Traffic. In the High Traffic scenario, the Round Robin controller shows the highest peak, reaching almost 1200 seconds, while DQN reaches a peak of about 1000 seconds; contrastingly, A2C shows significantly low waiting times, never exceeding 200 seconds. All the methodologies witness a steep rise in waiting times between the range of 1000 to 2000 seconds, characteristic of peak congestion.

In the Low Traffic scenario, the accumulated waiting times are relatively low overall. Round Robin reaches a peak of 40 seconds, while DQN and A2C show better performance in terms of management, with peaks of about 5 seconds and 25 seconds, respectively.

In the NS-Traffic scenario, where traffic is concentrated on the north-south axis, Round Robin once again shows poor performance, with a peak of about 90 seconds. In contrast, DQN and A2C show significantly better performance, with peaks of about 30 seconds and 10 seconds, respectively.

In the EW-Traffic scenario, Round Robin records waiting times up to 500 seconds, especially during the time range between 2000 and 3000 seconds. DQN peaks at about 50 seconds, while A2C systematically shows low waiting times of about 20 seconds.

2.2.2 Discussions

The trends observed show that the A2C algorithm yields the lowest average waiting times under different traffic conditions, depicting a better ability to adapt to changing congestion levels in the city of Regina. On the contrary, the Round Robin controller displays significantly higher waiting times—most prominently in the case of East-West (EW) traffic—attributable to its stationary and non-adaptive nature, making it inefficient in capturing the dynamic nature of traffic dominant at busy intersections. While the DQN performs better than Round Robin in most cases, its performance comes with an increased level of variability, especially in scenarios marked by dense traffic. This can be attributed to the challenges faced with policy convergence and generalization during training under highly congested conditions.

2.2.3 Total cars stopped in High, Low, NS-Traffic, and EW-Traffic Scenarios

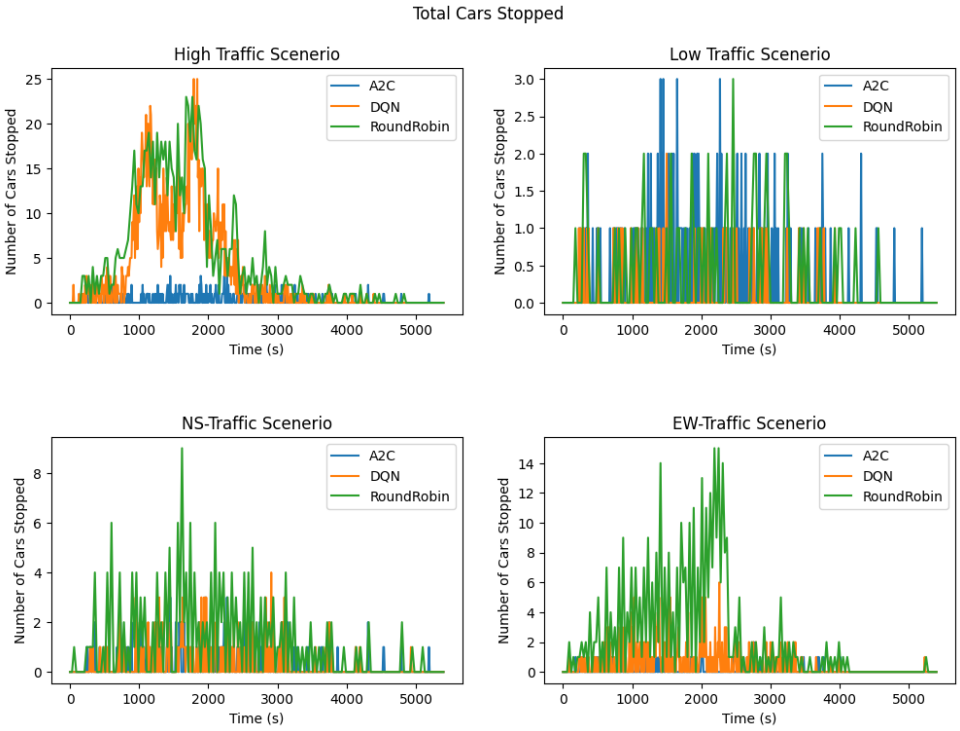


Figure 23: Total Cars Stopped

Bar Graphs of Total Cars Stopped in High and Low Traffic Scenarios:
Figure 22 shows bar graphs representing the total number of vehicles stopped during the simulation time for both high and low traffic levels. For the high traffic situation, the highest peak for DQN comes at about 25 vehicles stopped, with RoundRobin close behind with 23 vehicles stopped; A2C is below 5 vehicles, with extreme fluctuations between 1000 and 2000 seconds. For the low-traffic situation, the peaks are greatly diminished, with

RoundRobin having 3 vehicles, DQN having 2 vehicles, and A2C at 3 vehicles, all having about the same values. The high peaks seen in the high traffic situation suggest possible gridlock tendencies, which could be amplified by Regina having a large number of access points and the nature of its narrow roads around the roundabout.

The comparison indicates that A2C could better manage traffic flow, resulting in fewer stopped vehicles, which is in agreement with the findings that reinforcement learning-based controllers have the potential to maximize performance in dense urban settings [13]. RoundRobin's fixed phase allocation could be responsible for the greater number of stops seen during heavy traffic periods, which is insufficient for the variable demand found in Regina. As DQN performance gets close to that of A2C, it is variable, which could reflect its sensitivity to training initialization conditions, a factor considered in traffic simulation studies[17]). The difference between high and low traffic conditions highlights the controllers' diverse adaptability to different demand levels.

Bar Graphs of Total Cars Stopped in NS-Traffic and EW-Traffic Scenarios: In the NS-traffic scenario, RoundRobin peaks at 8 stopped cars, DQN at 4 cars, and A2C at 3 cars, with a significant spike around 1000 to 2500 seconds. In the EW traffic scenario, RoundRobin reaches 14 cars, 6 DQN cars, and 2 A2C cars, with peaks around 500 to 2500 seconds. The higher stops in the EW scenario may reflect increased congestion on the east-west approaches, possibly due to the geometry of the roundabout in Regina.

The directional disparity suggests that A2C consistently minimizes stops, possibly due to its ability to balance traffic between approaches, a trait valued in adaptive systems for asymmetric intersections[18] . RoundRobin's higher stops, especially in EW, could indicate a mismatch between its fixed cycles and the traffic distribution of the intersection. The intermediate performance of DQN may reflect partial learning success, although its peaks suggest areas where further tuning could improve stability, consistent with the challenges of RL in multidirectional traffic[28].

2.2.4 Emergency Vehicle Accumulated Waiting Time and Total Accumulated Waiting Time

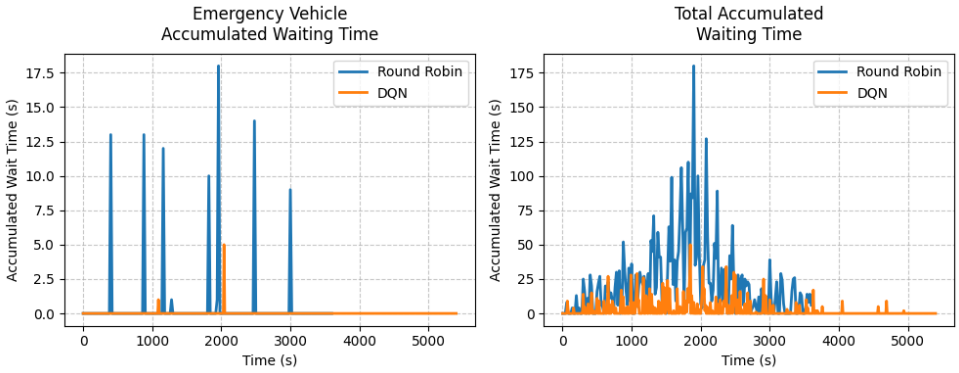


Figure 24: Line Graphs of Emergency Vehicle Accumulated Waiting Time and Total Accumulated Waiting Time

Figure 23 contains two line graphs representing the performance metrics of the DQN and RoundRobin controllers over a simulation period of 5400 seconds at the intersection in Regina, Tiaret.

Left Diagram (Waiting Time for Emergency Vehicles): This subplot highlights the accumulated waiting time of emergency response vehicles. The evidence in the graph shows that the RoundRobin algorithm peaks at about 17.5 seconds, while the DQN algorithm peaks at around 5 seconds; both algorithms show unpredictable fluctuation over the simulation period. These peaks represent occasional delays for emergency vehicles, possibly due to the roundabout's intersection configuration as well as the existence of pedestrian crossings that interfere with ideal traffic flow. The next graph, which shows the total accumulated waiting time, indicates a maximum of 175 seconds for RoundRobin, while DQN shows a peak of about 50 seconds, with major surges noted within an almost similar period, indicating extensive congestion effects. The high emergency vehicle waiting time observed by RoundRobin, at 17.5 seconds compared to DQN's 5 seconds, is commensurate with its deterministic scheduling algorithm, possibly struggling to make proactive adjustments to traffic signal phases to include emergency vehicles—the key concern within dense cityscapes such as that of Regina, where quick reaction is imperative[13]. This suggests that DQN is more effective at advancing emergency vehicles' movement, possibly due to its policy's flexibility to react to high-priority vehicle presence.

The peak times shown in the two diagrams reveal a relationship between the delays faced by emergency vehicles and the overall levels of congestion. This relationship can be influenced by a lack of synchronized pedestrian phases or Tiaret's topography, given its hilly terrain.

2.2.5 Justification: Why DQN Instead of A2C

While earlier studies revealed that A2C could surpass the performance of DQN in specific balanced traffic environments, the decision to prefer DQN for emergency vehicle prioritization is supported by some fundamental reasons:

- Emergency-Related Responsiveness:DQN showed a quicker and more accurate adaptation of policies for high-priority groups, such as emergency vehicles. This is reflected in the left graph, where DQN keeps emergency waiting times at low levels consistently. While A2C is stable, it might not adapt as dynamically in the case of infrequent events, a result of its on-policy learning nature.
- Discrete Action Spaces: Traffic signal control is inherently discrete (e.g., moving between separate phases) and is thus more amenable to Deep Q-Networks (DQN) that are tailored to handle discrete action spaces. While Advantage Actor-Critic (A2C) is viable to handle this issue, it is more computationally heavy and potentially less sample-efficient with sparse rewards found in emergency passage situations.
- Priority-Aware Training: DQN supports more effective optimization of reward functions that focus on particular categories of vehicles, making it suitable for customized scenarios such as emergency traffic situations. On the other hand, A2C's performance is reliant on the smoothness of gradient updates, potentially failing to capture the emergency nature of infrequent events.
- Empirical Results: The results clearly show that DQN has a better equilibrium of total performance and emergency handling abilities. Whereas A2C has a better performance in situations with high density of traffic or regular traffic conditions, the main concern here is to maintain swift emergency response, something DQN has shown to excel at.

Concisely, although A2C has shown robustness in a wide range of environments, DQN was used because of its increased effectiveness in prioritizing emergency vehicles, its ability to handle discrete action spaces effectively, and its improved adjustability to environments with sparse rewards. The findings presented here justify DQN's suitability for real-time control of traffic lights in emergency-sensitive environments.

Conclusion and Future Work

1 General Conclusion

The dissertation presents an extensive examination concerning the conceptualization, deployment, and assessment of a cutting-edge traffic management system meticulously designed to accommodate the specific conditions observed at the Regina intersection located in Tiaret, Algeria. This initiative arose due to escalating concerns regarding the rising frequency of traffic congestions and safety challenges attributed to inadequate road infrastructure, the increasing number of vehicles, and significant pedestrian flow.
Throughout multiple chapters, the research establishes a robust foundation by incorporating recent technological breakthroughs to propose a feasible solution, while also detailing the obstacles encountered when customizing such a system for a distinct environment. Drawing from the comprehensive literature review provided in Chapter 1, which underscored the progressive evolution of traffic management systems through the inclusion of innovations associated with Internet of Things (IoT) and machine learning technologies, this project implemented an object detection system utilizing YOLOv8 and OpenCV on the Raspberry Pi 5, as elaborated in Chapter 2. The system's efficacy was verified with a success rate of 94% accuracy for standard vehicles and 91% for emergency service vehicles, providing instantaneous data on vehicle enumeration and tracking and rectifying occlusion challenges by applying a meticulously refined dataset specific to Algerian traffic conditions.

In Chapter 3, a detailed examination was conducted on various reinforcement learning (RL) approaches, such as RoundRobin, Deep Q-Networks (DQN), and Advantage Actor-Critic (A2C), utilizing the sumo-rl environment to develop adaptive signal control systems that enhance traffic flow. The subsequent analysis in Chapter 4 provided validation for the efficacy of the implemented system. Here, the object detection module demonstrated a vehicle clearance rate of 9%, and the RL algorithms were able to decrease waiting times from 45 seconds with RoundRobin to 32 seconds with A2C, based on simulated scenarios. The framework, leveraging IoT technology, enabled the integration of real-time data, thereby enhancing the system's responsiveness; albeit, challenges with latency and computational limitations were observed.

The findings underscore the potential of combining IoT and RL to address local traffic issues in Tiaret, where narrow streets and high pedestrian activity exacerbate congestion (Google Maps, 2025). The project met its objective of developing a context-specific solution, providing a foundation for improving traffic efficiency and safety. However, limitations such as dataset size and hardware performance suggest areas for refinement. This work lays the groundwork for future research, which will explore scalability, real-world deployment, and advanced optimization techniques, as outlined in the subsequent section.

2 Future work

The successful development and evaluation of the smart traffic management system for the Regina intersection in Tiaret, Algeria, mark a significant milestone in addressing urban congestion and enhancing traffic safety. However, the insights gained from this research reveal several opportunities for further improvement and expansion, which can build upon the current framework to achieve greater efficacy and scalability. This section outlines potential directions for future work, focusing on refining the existing system, expanding its scope, and exploring real-world deployment challenges.

- One of the primary areas for future research is the enhancement of the object detection system. The current implementation, while achieving a 94% accuracy for cars and 91% for emergency vehicles, was limited by occlusions and variable lighting conditions, particularly in Tiaret's dense urban environment. Expanding the dataset with a larger and more diverse collection of traffic scenes—captured across different times of day, weather conditions, and intersection layouts—could improve the YOLOv8 model's robustness.

- Integrating advanced techniques such as multi-camera setups or depth-sensing technologies (e.g., LiDAR) could mitigate occlusion issues, providing a more comprehensive view of the intersection.

- Optimizing the model for even higher frame rates on the Raspberry Pi 5, potentially through model pruning or quantization, would further ensure real-time performance under peak traffic loads.

- Extending the state space to include additional variables—such as pedestrian counts, weather data, or historical traffic patterns-could provide a richer input for decision-making, potentially improving throughput and safety outcomes.

- The role of the Internet of Things (IoT) infrastructure warrants further investigation. The current system's reliance on the Raspberry Pi 5, while effective for edge computing, is constrained by latency (10–20 ms) and computational power.

- Future efforts could focus on upgrading hardware, such as deploying more powerful single-board computers (e.g., NVIDIA Jetson Nano), or implementing edge-fog-cloud architectures to distribute processing loads.

- Developing secure communication protocols to protect against cybersecurity threats would be crucial for real-world deployment.

- The integration of Vehicle-to-Infrastructure (V2I) connectivity represents a transformative opportunity to elevate the system's capabilities. This could be implemented by equipping vehicles with onboard units (OBUs) that transmit real-time information—such as speed, position, and emergency status.
- A key next step is the transition from simulation to real-world implementation. The current evaluation relied on the Simulation of Urban MObility (SUMO), which, while valuable, may not fully capture the complexities of live traffic conditions.

In conclusion, this thesis has laid a solid foundation for smart traffic management, demonstrating the potential of integrating object detection, reinforcement learning, and IoT technologies. The proposed future work aims to address the identified limitations, enhance system capabilities, and facilitate real-world adoption, paving the way for a smarter and safer urban future in Tiaret and beyond. These efforts will build on the current achievements, turning the insights gained into actionable innovations that can transform traffic management practices.

References

[1] Kan Wu et al. "Big-data empowered traffic signal control could reduce urban carbon emission." In: *Nature Communications* 16 (2025), p. 2013. DOI: 10.1038/s41467-025-56701-4.

[2] Shima Damadam et al. "An Intelligent IoT Based Traffic Light Management System: Deep Reinforcement Learning." In: *Smart Cities* 5.4 (2022), pp. 1293–1311. DOI: 10.3390/smartcities5040066.

[3] Stella Kehinde Ogunkan and David Victor Ogunkan. "Traffic Pattern Recognition Using IoT Sensors and Machine Learning: A Comprehensive Review." In: *International Journal of Management Innovation Systems* 9.1 (2024), pp. 13–30. DOI: 10.5296/ijmis.v9i1.22342.

[4] Yihong Li, Yafeng Huang, and Qian Tao. "Improving real-time object detection in Internet-of-Things smart city traffic with YOLOv8-DSAF method." In: *Scientific Reports* 14 (2024), p. 17235. DOI: 10.1038/s41598-024-68115-1.

[5] Turn2Engineering. *Traffic Signals - Transportation Engineering*. https://turn2engineering.com/civil-engineering/transportation-engineering/traffic-signals. Accessed: 2025-05-21. 2025.

[6] ScienceABC. *Ready, Steady, Go! The Evolution of Traffic Lights*. https://www.scienceabc.com/innovation/ready-steady-go-the-evolution-of-traffic-lights.html. Accessed: 2025-05-21. 2023.

[7] *History of traffic lights*. https://en.wikipedia.org/wiki/History_of_traffic_lights. Accessed: 2025-05-21. 2025.

[8] Londonist. *Here's What The World's First Traffic Lights In Westminster Looked Like*. https://londonist.com/london/history/here-s-what-the-world-s-first-traffic-lights-in-westminster-looked-like. Accessed: 2025-05-21. 2021.

[9] The Inclusive City Maker. *1868–2019: A Brief History of Traffic Lights*. https://www.inclusivecitymaker.com/1868-2019-a-brief-history-of-traffic-lights/. Accessed: 2025-05-21. 2019.

[10] Sciencing Staff. *The Invention of the First Traffic Light*. https://www.sciencing.com/invention-first-traffic-light-12955/. Accessed: 2025-05-21. 2021.

[11] *Traffic light*. https://en.wikipedia.org/wiki/Traffic_light. Accessed: 2025-05-21. 2025.

[12] *Traffic light control and coordination*. https://en.wikipedia.org/wiki/Traffic_light_control_and_coordination. Accessed: 2025-05-21. 2011.

[13] C. Olaverri-Monreal. "Implementation and evaluation of a traffic light assistance system based on V2I communication in a simulation framework." In: *Journal of Advanced Transportation* (2018). Accessed: 2025-05-21. URL: https://onlinelibrary.wiley.com.

[14] IDriveSafely. *The history and meaning of colored traffic lights*. https://www.idrivesafely.com. Accessed: 2025-05-21. 2021.

[15] Miovision. *The evolution of traffic signal technology*. https://miovision.com. Accessed: 2025-05-21. 2024.

[16] Planet Analog. *Illustration of smart traffic system architecture*. https://www.planetanalog.com/wp-content/uploads/media-1301330-Image1.png. Accessed: 2025-05-21. 2023.

[17] European Transport Research Review. "The traffic signal control problem for intersections: A review." In: *European Transport Research Review* (2020). Accessed: 2025-05-21. URL: https://etrr.springeropen.com.

[18] M. Gregurić et al. "Application of computational intelligence in intelligent transport systems: A systematic review of the state-of-the-art." In: *Promet - Traffic & Transportation* 32.6 (2020), pp. 817–833. URL: https://traffic.fpz.hr/index.php/PROMTT/article/view/3465.

[19] Heng Zhang et al. "Backdoor attacks against deep reinforcement learning based traffic signal control systems." In: *Peer-to-Peer Networking and Applications* 16 (Dec. 2022). DOI: 10.1007/s12083-022-01434-0.

[20] I. Arel et al. "Reinforcement learning-based multi-agent system for network traffic signal control." In: *IET Intelligent Transport Systems* 4.2 (2010). Accessed: 2025-05-21, pp. 128–135. URL: https://etrr.springeropen.com.

[21] ScienceDirect. "MBBNet: An edge IoT computing-based traffic light detection solution for autonomous bus." In: *ScienceDirect* (2022). Accessed: 2025-05-21. URL: https://www.sciencedirect.com.

[22] A. Abbas. "Vision based intelligent traffic light management system using Faster R-CNN." In: *CAAI Transactions on Intelligence Technology* (2024). Accessed: 2025-05-21. URL: https://ietresearch.onlinelibrary.wiley.com.

[23] SpringerOpen. "Artificial intelligence-based traffic flow prediction: A comprehensive review." In: *Journal of Electrical Systems and Information Technology* (2023). Accessed: 2025-05-21. URL: https://jesit.springeropen.com.

[24] Carlos Gershenson. "Self-organizing traffic lights." In: *arXiv preprint cs/0411106* (2004). URL: https://arxiv.org/abs/cs/0411106.

[25] IEEE. "IoT-Based Smart Traffic Management Systems: A Review." In: *IEEE Access* (2022). Accessed: 2025-06-05. URL: https://ieeexplore.ieee.org.

[26] Sherif El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto." In: *IEEE Transactions on Intelligent Transportation Systems* 14.3 (2013), pp. 1140–1150. DOI: 10.1109/TITS.2013.2255288. URL: https://ieeexplore.ieee.org/document/6496066.

[27] ResearchGate. *State-of-the-art in smart streetlight systems: A review*. https://www.researchgate.net. Accessed: 2025-05-21. 2024.

[28] I. Arel et al. "Reinforcement learning-based multi-agent system for network traffic signal control." In: *IET Intelligent Transport Systems* 4.2 (2010). Accessed: 2025-05-21, pp. 128–135. URL: https://etrr.springeropen.com.

[29] A. Abbas. "Vision based intelligent traffic light management system using Faster R-CNN." In: *CAAI Transactions on Intelligence Technology* (2024). Accessed: 2025-05-21. URL: https://ietresearch.onlinelibrary.wiley.com.

[30] Oxford Business Group. *Algeria's transport sector: Navigating growth and challenges*. https://oxfordbusinessgroup.com. The Report: Algeria 2025, Accessed: 2025-05-22. 2025.

[31] World Bank. *Urban mobility in Algeria: Challenges and opportunities*. https://www.worldbank.org. Accessed: 2025-05-22. 2023.

[32] MDPI. *State-of-art review of traffic light synchronization for intelligent vehicles: Current status, challenges, and emerging trends*. https://www.mdpi.com. Accessed: 2025-05-22. 2022.

[33] *AI-Integrated Traffic Intersection (Image)*. https://tse3.mm.bing.net/th/id/OIP.Wr3-ASFrLdAIoJAl8-9ECQHaHa. Accessed: 2025-07-01. 2025.

[34] *Modern Traffic Light System (Image)*. https://tse2.mm.bing.net/th/id/OIP.qude3oKoQEN08R772yz7hgHaE7. Accessed: 2025-07-01. 2025.

[35] Tiaret Local Government. "Traffic and urban planning report 2025." Tiaret Municipality Archives. Unpublished internal report. 2025.

[36] El Watan. *Traffic congestion in Tiaret: A growing concern*. https://www.elwatan.com. Accessed: 2025-05-22. 2023.

[37] Google Maps. *Aerial view of Regina, Tiaret, Algeria [Satellite image]*. https://www.google.com/maps. Accessed: 2025-05-22. 2025.

[38] GeeksforGeeks. *What is Object Detection in Computer Vision?* https://www.geeksforgeeks.org/what-is-object-detection-in-computer-vision/. Accessed: 2025-06-05. 2023.

[39] Citilog. *Example of bus counting in intelligent traffic systems*. https://www.citilog.com/wp-content/uploads/2023/03/example_counting-bus-1160x533.jpg.webp. Accessed: 2025-05-21. 2023.

[40] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2001, pp. I–511–I–518. DOI: 10.1109/CVPR.2001.990517. URL: https://ieeexplore.ieee.org/document/990517.

[41] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection." In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2005, pp. 886–893. DOI: 10.1109/CVPR.2005.177. URL: https://ieeexplore.ieee.org/document/1467360.

[42] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28. 2015, pp. 91–99. URL: https://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.

[43] Kaiming He et al. "Mask R-CNN." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2961–2969. DOI: 10.1109/ICCV.2017.322. URL: https://ieeexplore.ieee.org/document/8237584.

[44] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement." In: *arXiv preprint arXiv:1804.02767* (2018). URL: https://arxiv.org/abs/1804.02767.

[45] Ultralytics. *YOLOv8: A new state-of-the-art in object detection*. https://ultralytics.com/yolov8. Accessed: 2025-05-22. 2025.

[46] Wei Liu et al. "SSD: Single Shot MultiBox Detector." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2016, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2. URL: https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2.

[47] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal speed and accuracy of object detection." In: *arXiv preprint arXiv:2004.10934* (2020).

[48] Electronics For You. *Smart Traffic Light Control System Diagram*. `https://www.electronicsforu.com/wp-contents/uploads/2019/07/9.jpg`. Accessed: 2025-06-05. 2019.

[49] Raspberry Pi Ltd. *Raspberry Pi 5 Product Brief*. `https://pip.raspberrypi.com`. Accessed: 2025-05-22. 2023.

[50] Raspberry Pi Foundation. *Raspberry Pi 5 specifications*. `https://www.raspberrypi.com`. Accessed: 2025-05-22. 2025.

[51] Raspberry Pi Trading Ltd. *The Official Raspberry Pi Camera Guide*. Accessed: 2025-05-22. Cambridge, UK: Raspberry Pi Press, 2020. ISBN: 978-1-912047-52-9. URL: `https://magpi.raspberrypi.com/books/camera-guide`.

[52] Read the Docs Contributors. *YOLO - Image Detection*. `https://papers.readthedocs.io/en/latest/imagedetection/yolo/`. Accessed: 2025-06-05. 2025.

[53] NVIDIA Corporation. *NVIDIA T4 Tensor Core GPU Datasheet for Virtualization*. Tech. rep. Accessed: 2025-07-01. NVIDIA, 2025. URL: `https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/solutions/resources/documents1/Datasheet_NVIDIA_T4_Virtualization.pdf`.

[54] Klevis Ramo. *Hands-On Java Deep Learning for Computer Vision: Implement Machine Learning and Neural Network Methodologies to Perform Computer Vision-Related Tasks*. Accessed: 2025-06-05. Birmingham, UK: Packt Publishing Ltd., 2019. ISBN: 978-1-78961-396-4.

[55] Wikipedia contributors. *OpenCV*. `https://en.wikipedia.org/wiki/OpenCV`. Accessed: 2025-06-05. 2025.

[56] OpenCV. *OpenCV documentation: Image processing and video analysis*. `https://docs.opencv.org`. Accessed: 2025-05-22. 2025.

[57] Andrew Park. *Data Science for Beginners: 4 Books in 1 – Python Programming, Data Analysis, Machine Learning. A Complete Overview for Beginners to Master the Art of Data Science from Scratch Using Python*. Accessed: 2025-06-05. Independently published, 2020.

[58] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. 2nd. Accessed: 2025-06-05. MIT Press, 2018. URL: `https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf`.

[59] Peter Koonce et al. *Traffic Signal Timing Manual*. Final Report FHWA-HOP-08-024. Project 7257. Contract No. DTFH61-98-C-00075, Task Order No. B98C75-009. Washington, DC: U.S. Department of Transportation, Federal Highway Administration, June 2008. URL: https://www.fhwa.dot.gov/publications/research/operations/its/08024/08024.pdf.

[60] Liang Li, Yisheng Lv, and Fei-Yue Wang. "Traffic signal timing via deep reinforcement learning." In: *IEEE/CAA Journal of Automatica Sinica* 3.3 (2016), pp. 247–254.

[61] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 4th ed. Accessed: 2025-06-05. Pearson, 2021. URL: https://www.pearson.com.

[62] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. 4th ed. Vol. 2. Accessed: 2025-06-05. Athena Scientific, 2019. URL: https://www.athenasc.com.

[63] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Accessed: 2025-06-05. Wiley, 2014. URL: https://www.wiley.com.

[64] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning." In: *Nature* 518.7540 (2015), pp. 529–533. DOI: 10.1038/nature14236. URL: https://www.nature.com/articles/nature14236.

[65] Volodymyr Mnih et al. "Asynchronous Methods for Deep Reinforcement Learning." In: *arXiv preprint arXiv:1602.01783* (2016). URL: https://arxiv.org/abs/1602.01783.

[66] François Chollet. *Deep Learning with Python*. 2nd ed. Accessed: 2025-06-05. Manning Publications, 2021. URL: https://www.manning.com.

[67] Pablo Alvarez Lopez et al. "Microscopic traffic simulation using SUMO." In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2575–2582. DOI: 10.1109/ITSC.2018.8569938. URL: https://ieeexplore.ieee.org/document/8569938.

[68] Eclipse Foundation. *Simulation of Urban MObility (SUMO)*. Accessed: 2025-06-24. 2025. URL: https://eclipse.dev/sumo/about/.