A logo with text on it

AI-generated content may be incorrect.A logo of a university

AI-generated content may be incorrect.

**UNIVERSITÉ CHOUAÏB DOUKKALI**

**Faculté des sciences El-Jadida**

**Département d’informatique**

**PROJET DE FIN D'ÉTUDES**

**Filière : SMI**

**Titre du rapport :**

**Simulation et Optimisation du Trafic Routier à l’Aide d’Agents Intelligents**

Présenté par : **NAJI YOUSSEF**

Encadré par : **M. IDRISS ABADA**

Soutenu le : **23/06/2025**

Lieu : **Faculté de sciences El-Jadida**

Année universitaire : 2024 – 2025

Projet de Fin d’Études

Titre : Système intelligent de gestion du trafic par apprentissage par renforcement

Remerciements

Résumé

Abstract

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations

Introduction générale

Chapitre 1 : Contexte et bases théoriques

1.1 Introduction

1.2 Problématique du trafic urbain

1.3 Apprentissage automatique et par renforcement

1.4 Modèle MDP et algorithmes de base

1.5 Conclusion

Chapitre 2 : État de l’art

2.1 Introduction

2.2 Travaux basés sur Q-learning

2.3 Travaux sur DQN et ses variantes

2.4 MARL et QMIX dans le trafic intelligent

2.5 Limites des approches existantes

2.6 Conclusion

Chapitre 3 : Conception générale du système

3.1 Introduction

3.2 Objectifs et contraintes

3.3 Outils et bibliothèques utilisés (SUMO, PyTorch, TraCI, etc.)

3.4 Modélisation de l’intersection dans SUMO

3.5 Définition des états, actions et récompenses

3.6 Architecture globale du système

// Cette section présente l’architecture générale du système intelligent de gestion du trafic, en mettant en évidence les interactions entre les différents outils et modules utilisés.

3.7 Conclusion

Chapitre 4 : Q-learning appliqué à la gestion du trafic

4.1 Introduction

4.2 Implémentation du Q-learning

4.3 Résultats expérimentaux

4.4 Analyse et limites

4.5 Conclusion

Chapitre 5 : Deep Q-Network (DQN)

5.1 Introduction

5.2 Implémentation du DQN

5.3 Résultats expérimentaux

5.4 Analyse et comparaison avec Q-learning

5.5 Conclusion

Chapitre 6 : Multi-Agent et QMIX

6.1 Introduction

6.2 Implémentation de QMIX

6.3 Résultats expérimentaux

6.4 Comparaison globale des approches

6.5 Limites et perspectives

6.6 Conclusion

Conclusion générale

Bibliographie

**Remerciements**

Avant toute chose, je rends grâce à Dieu, le Tout-Puissant, pour m’avoir accordé la force, la patience et la persévérance nécessaires pour mener à bien ce travail.

Je tiens à exprimer ma profonde gratitude à **M. IDRISS ABADA** , mon encadrant, pour sa disponibilité, ses conseils avisés, son accompagnement constant et la richesse de ses orientations tout au long de ce projet. Son expertise et son engagement ont grandement contribué à l’aboutissement de ce travail.

Mes sincères remerciements s’adressent également aux membres du **jury**, pour l’intérêt qu’ils ont porté à ce travail, le temps qu’ils lui ont consacré et les remarques constructives qu’ils ne manqueront pas d’apporter.

Je souhaite également remercier du fond du cœur **mes parents**, pour leur soutien moral et matériel tout au long de mes études, ainsi que **ma petite famille** et **mes amis**, pour leur présence, leurs encouragements et leur confiance tout au long de ce parcours académique.

**Résumé**

Le transport routier joue aujourd’hui un rôle fondamental dans le fonctionnement des sociétés modernes. Toutefois, la gestion efficace du trafic constitue un défi majeur, en particulier dans les zones urbaines fortement congestionnées. Les feux de signalisation, bien qu’essentiels, fonctionnent souvent selon des cycles fixes, ce qui limite leur efficacité face aux fluctuations du trafic.

Ce projet vise à développer une solution intelligente de contrôle dynamique des feux de signalisation, reposant sur l’**apprentissage par renforcement** combiné à l’**apprentissage profond**. Grâce à cette approche, le système proposé est capable de s’adapter aux conditions réelles de circulation en temps réel, en prenant des décisions optimisées pour améliorer la fluidité du trafic.

L’objectif final est de concevoir un agent intelligent qui apprend à ajuster les phases des feux de manière autonome, dans un environnement simulé, afin de réduire les files d’attente, les temps d’arrêt et les embouteillages.

**Abstract**

Road transport plays a vital role in the functioning of modern societies. However, efficiently managing traffic remains a major challenge, especially in densely populated urban areas. Although traffic lights are essential for regulating flow, they often operate with fixed-time cycles, limiting their adaptability to real-time traffic conditions.

This project aims to develop an intelligent solution for **dynamic traffic light control** using **reinforcement learning** combined with **deep learning**. This approach allows the system to adapt in real time to actual traffic situations and make optimized decisions to improve traffic flow.

The ultimate goal is to design an intelligent agent capable of autonomously adjusting signal phases within a simulated environment, in order to reduce queues, waiting times, and congestion.

**Table des matières**

 **Remerciements**

 **Résumé**

 **Abstract**

 **Table des matières**

 **Liste des figures**

[Figure 1 Traffic Intersection Simulation 25](#_Toc200808357)

[Figure 2 Simulation d’une double intersection de trafic 25](#_Toc200808358)

[Figure 3 Q-Leanrnig Architicture 28](#_Toc200808359)

 **Liste des tableaux**

 **Liste des abréviations**

 **Introduction générale**

**Chapitre 1 : Contexte et bases théoriques**

**1.1 Introduction  
1.2 Problématique du trafic urbain  
1.3 Apprentissage automatique et par renforcement  
1.4 Modèle MDP et algorithmes de base  
1.5 Conclusion**

**Chapitre 2 : État de l’art**

**2.1 Introduction  
2.2 Travaux basés sur Q-learning  
2.3 Travaux sur DQN et ses variantes  
2.4 MARL et QMIX dans le trafic intelligent  
2.5 Limites des approches existantes  
2.6 Conclusion**

**Chapitre 3 : Conception générale du système**

**3.1 Introduction  
 3.2 Objectifs et outils utilisés**

**3.3 Modélisation des intersections**

**3.4 Gestion du feu jaune dynamique**

**3.5 États, actions et recompenses**

**3.6 Architecture globale du système**

**3.7 Conclusion**

**Chapitre 4 : Q-learning appliqué à la gestion du trafic**

**4.1 Introduction  
4.2 Implémentation du Q-learning  
4.3 Résultats expérimentaux  
4.4 Analyse et limites  
4.5 Conclusion**

**Chapitre 5 : Deep Q-Network (DQN)**

**5.1 Introduction  
5.2 Implémentation du DQN  
5.3 Résultats expérimentaux  
5.4 Analyse et comparaison avec Q-learning  
5.5 Conclusion**

**Chapitre 6 : Multi-Agent et QMIX**

**6.1 Introduction  
6.2 Implémentation de QMIX  
6.3 Résultats expérimentaux  
6.4 Comparaison globale des approches  
6.5 Limites et perspectives  
6.6 Conclusion**

**Conclusion générale**

**Bibliographie**

**Introduction générale**

Dans un contexte de croissance urbaine rapide et d’augmentation constante du nombre de véhicules, la gestion du trafic routier est devenue un enjeu majeur pour les villes modernes. Les embouteillages, les temps d’attente prolongés aux intersections et la pollution engendrée par la circulation sont autant de problématiques qui affectent non seulement l’efficacité des déplacements urbains, mais aussi la qualité de vie des citoyens.

Traditionnellement, la régulation du trafic repose sur des systèmes à base de feux de signalisation programmés selon des cycles fixes. Ces systèmes, bien qu’essentiels, présentent des limites notables puisqu’ils ne s’adaptent pas en temps réel aux conditions variables du trafic. Ce manque de flexibilité peut entraîner une mauvaise coordination des flux de véhicules, accentuant la congestion et les pertes de temps.

Face à ces défis, les avancées technologiques récentes, notamment dans le domaine de l’intelligence artificielle, offrent de nouvelles perspectives prometteuses. En particulier, l’apprentissage par renforcement (Reinforcement Learning, RL) et ses variantes plus récentes comme le Deep Q-Learning (DQN) ou encore les approches multi-agents (MARL, QMIX), permettent de concevoir des systèmes intelligents capables d’apprendre à prendre des décisions optimales en fonction de l’environnement.

Dans ce projet, nous proposons de développer un système intelligent de gestion du trafic basé sur ces techniques, en utilisant un environnement de simulation réaliste via SUMO (Simulation of Urban MObility), couplé à des outils tels que TraCI et PyTorch. L’objectif est de concevoir un agent (ou un ensemble d’agents) capable de contrôler dynamiquement les feux de signalisation à une intersection, de manière à fluidifier la circulation, réduire les files d’attente et optimiser le temps de passage.

Ce rapport est structuré comme suit :

* Le Chapitre 1 introduit les bases théoriques de l’apprentissage automatique et par renforcement, en posant les fondements nécessaires à la compréhension de notre démarche.
* Le Chapitre 2 présente un état de l’art des travaux existants dans le domaine du contrôle intelligent du trafic, en mettant l’accent sur les approches basées sur le Q-learning, le DQN et les systèmes multi-agents.
* Le Chapitre 3 détaille la conception générale du système proposé, les outils utilisés ainsi que la modélisation de l’environnement de simulation.
* Les Chapitres 4, 5 et 6 sont respectivement consacrés à l’implémentation et à l’analyse des performances de Q-learning, DQN et QMIX appliqués à notre problème.
* Enfin, une conclusion générale viendra clore ce rapport en résumant les résultats obtenus et en proposant des perspectives d’amélioration futures.

**Chapitre 1 : Contexte et bases théoriques**

* 1. **Introduction**

La régulation du trafic urbain est un défi de plus en plus complexe, notamment dans les grandes agglomérations. Le développement démographique, l’augmentation du nombre de véhicules et les déplacements quotidiens rendent les infrastructures routières souvent insuffisantes face à la demande croissante. Dans ce contexte, les systèmes traditionnels de gestion du trafic, tels que les feux de signalisation à cycle fixe, deviennent obsolètes et inefficaces. Il est donc nécessaire d’envisager des solutions intelligentes, capables de s’adapter dynamiquement à l’état du trafic en temps réel.

Ce chapitre présente les fondements nécessaires pour comprendre les techniques utilisées dans ce projet, à savoir l’**apprentissage automatique**, l’**apprentissage par renforcement**, ainsi que le **modèle de décision de Markov (MDP)** et les algorithmes de base qui en découlent.

* 1. **Problématique du trafic urbain**

La croissance rapide des zones urbaines et l’augmentation du parc automobile ont engendré une pression sans précédent sur les réseaux routiers. Selon des études récentes, les embouteillages dans les grandes villes entraînent des pertes de temps cumulées de plusieurs heures par jour pour les usagers, une hausse des émissions de CO2 et des coûts économiques se chiffrant en milliards. Les causes principales incluent la densité croissante des véhicules, les infrastructures souvent sous- imprévues comme les accidents ou les travaux. Les systèmes traditionnels, tels que les feux à cycle fixe, ne peuvent s’adapter aux variations en temps réel, ce qui aggrave les retards et la congestion. La problématique centrale de ce projet peut ainsi se formuler comme suit : comment concevoir un système intelligent de gestion du trafic, capable de s’adapter dynamiquement aux conditions changeantes, afin de minimiser les temps d’attente, réduire la congestion et améliorer la fluidité des déplacements urbains ?

* 1. **Apprentissage automatique et par renforcement**

**a) Apprentissage automatique (Machine Learning)**

L'apprentissage automatique désigne un ensemble de techniques permettant à une machine d’apprendre à partir de données, sans être explicitement programmée. Il se divise principalement en trois catégories :

* Apprentissage supervisé
* Apprentissage non supervisé
* Apprentissage par renforcement (celui qui nous intéresse)

**b) Apprentissage par renforcement (Reinforcement Learning - RL)**

Contrairement au ML classique, l’apprentissage par renforcement repose sur un agent qui apprend à prendre des décisions optimales en interagissant avec un environnement. L’agent teste des actions, reçoit des retours sous forme de récompenses ou de pénalités, et ajuste sa stratégie pour maximiser la récompense à long terme. Dans la gestion du trafic, un agent RL peut, par exemple, ajuster la durée des feux de signalisation pour réduire les temps d’attente, en s’adaptant dynamiquement aux conditions en temps réel.

* 1. **Modèle MDP et algorithmes de base**

Le modèle de décision de Markov (MDP) constitue un cadre mathématique fondamental en apprentissage par renforcement, particulièrement bien adapté à des systèmes de décision séquentielle comme la gestion du trafic urbain.

Selon Puterman (1994), un MDP permet de modéliser un processus dans lequel un agent prend des décisions à chaque instant, dans un environnement incertain, avec pour objectif de maximiser une récompense cumulée au fil du temps. Il est défini par les éléments suivants :

* États (S) : Ils décrivent les différentes situations possibles de l’environnement. Dans le cas du trafic, un état peut représenter, par exemple, le nombre de véhicules en attente sur chaque voie, la densité de circulation ou l’état actuel des feux (vert, rouge, etc.).
* Actions (A) : Ce sont les décisions que l’agent peut prendre. Pour un carrefour, cela inclut des choix tels que prolonger un feu vert, passer à l’orange ou au rouge, ou ajuster dynamiquement la durée des cycles.
* Récompenses (R) : Elles mesurent la qualité de l’action prise dans un état donné. Une bonne politique cherchera à maximiser des critères comme la réduction du temps d’attente moyen, la diminution de la longueur des files, ou l’optimisation de la fluidité du trafic.
* Transitions (T) : Elles représentent les probabilités de passer d’un état à un autre après une action. Dans un contexte de trafic, les transitions sont stochastiques (aléatoires), car de nouveaux véhicules peuvent apparaître de manière imprévisible.

L’utilisation du MDP est particulièrement pertinente pour la gestion des carrefours intelligents, car il prend en compte à la fois :

* La séquentialité des décisions (chaque action influence l’état futur),
* Et la stochasticité du trafic (incertitude liée aux comportements des usagers de la route).

Le MDP fournit ainsi un cadre formel solide pour la modélisation du comportement d’un agent intelligent de contrôle de feux de signalisation.

* 1. **Conclusion**

Dans ce premier chapitre, nous avons exposé le contexte général lié à la gestion du trafic urbain, en soulignant les limites des systèmes de contrôle traditionnels, notamment ceux reposant sur des cycles de feux fixes.

Nous avons ensuite introduit les **principes fondamentaux de l’apprentissage automatique**, avec un accent particulier sur l’**apprentissage par renforcement**, qui offre un cadre flexible pour modéliser des décisions séquentielles dans un environnement incertain. Le **modèle de décision de Markov (MDP)** a été présenté comme un outil mathématique structurant permettant de formaliser les interactions entre un agent et son environnement dans un cadre stochastique.

Les notions d’états, d’actions, de récompenses et de transitions, propres au MDP, trouvent une application naturelle dans le contexte de la gestion du trafic, notamment pour modéliser le comportement d’un carrefour intelligent.

Ce socle théorique constitue la base sur laquelle s’appuiera le développement de notre système. Dans le chapitre suivant, nous explorerons les **travaux de recherche existants** relatifs à la gestion intelligente du trafic routier, en identifiant les approches, les outils et les limitations rencontrées dans la littérature.

**Chapitre 2 : État de l’art**

**2.1Introduction**

La régulation du trafic routier représente aujourd’hui un défi crucial pour les villes modernes, confrontées à une croissance continue du nombre de véhicules et à une saturation des infrastructures. Face à l’inefficacité des systèmes classiques de contrôle à temps fixe, les chercheurs se sont tournés vers des solutions intelligentes capables de s’adapter en temps réel aux conditions dynamiques de circulation.

L’émergence de l’intelligence artificielle, et plus particulièrement de l’**apprentissage par renforcement**, a ouvert la voie à de nouvelles approches dans le domaine de la gestion du trafic. Cette technique permet à un agent d’apprendre à prendre des décisions optimales par l’interaction avec l’environnement, sans supervision humaine directe.

Dans ce chapitre, nous présentons un **état de l’art des différentes méthodes** ayant été appliquées à la gestion des feux de signalisation. Nous abordons d’abord les **approches classiques**, puis les méthodes issues de l’**optimisation** et enfin, nous détaillons les techniques **d’apprentissage par renforcement**, qu’elles soient tabulaires (Q-learning), profondes (DQN) ou multi-agents (MARL, QMIX).

Cette revue critique permettra de situer notre travail dans le paysage scientifique actuel et de mettre en évidence les choix méthodologiques que nous adoptons dans la suite du projet.

**2.2.Approches classiques**

**2.2.1 Contrôle à temps fixe**

Les systèmes à cycles fixes reposent sur des durées prédéfinies pour chaque phase des feux (vert, rouge, orange). Bien que simples à implémenter, ils ne s'adaptent pas aux variations du trafic, entraînant des inefficacités significatives. Des études montrent une augmentation des temps d'attente moyens de 20 à 30 % dans des scénarios à forte variabilité [1].

**2.2.2. Règles adaptatives**

Les systèmes adaptatifs utilisent des capteurs (boucles magnétiques, caméras) pour ajuster les durées des feux selon des seuils prédéfinis, comme le nombre de véhicules détectés. Par exemple, le système SCATS prolonge la phase verte en fonction du volume de trafic [2]. Bien que plus flexibles, ces approches restent limitées par leur dépendance à des règles statiques, sans apprentissage basé sur l'expérience.

**2.3. Approches basées sur l'optimisation**

**2.3.1. Programmation mathématique**

La programmation linéaire et dynamique a été utilisée pour optimiser les séquences de feux, comme dans le modèle TRANSYT, qui minimise les retards à l'échelle d'un réseau [3]. Cependant, ces approches nécessitent une modélisation précise du trafic et sont coûteuses en calcul, limitant leur robustesse face aux aléas réels.

**2.3.2. Métahéuristiques**

Les métahéuristiques, telles que les algorithmes génétiques ou le recuit simulé, explorent l'espace des solutions pour optimiser la gestion des feux [4]. Efficaces pour des scénarios statiques, elles sont souvent limitées à des optimisations hors ligne en raison de leur complexité computationnelle.

**2.4. Apprentissage par renforcement (RL)**

L’apprentissage par renforcement (Reinfocement Learning-RL) est un type d’apprentissage automatique dans lequel un agent apprend à se comporter dans un environnement par essais et erreurs. L’agent reçoit des récompenses pour les actions qui mènent aux résultats souhaités et des pénalités pour les actions qui mènent aux résultats non souhaités. Au fil du temps, l’agent apprend à prendre des mesures qui maximisent ses récompenses [5] .

**2.4.1 Q-learning**

Le Q-learning est un algorithme hors-politique (off-policy) qui permet à un agent d’estimer la valeur d’une action dans un état donné. Il repose sur la mise à jour d’une Q-table associant chaque couple état-action à une valeur numérique, selon la formule de Bellman [6].

Dans le contexte de la gestion du trafic, les auteurs de [7] ont proposé un contrôleur de feux basé sur le Q-learning pour l’optimisation dynamique des phases de signalisation. Ce système s’adapte à l’état de la circulation en temps réel. En [8], une étude comparative entre Q-learning et SARSA a montré que le Q-learning est plus efficace dans des environnements stables à espace d’état modéré.

Cependant, lorsque le nombre d’états devient trop grand (intersections complexes, nombreuses voies), la Q-table devient ingérable, ce qui limite l’applicabilité du Q-learning aux réseaux urbains étendus.

**2.4.2 SARSA**

Le SARSA (*State-Action-Reward-State-Action*) est une méthode sur-politique (on-policy) dans laquelle l’agent apprend à partir de l’action réellement choisie, ce qui le rend plus prudent que le Q-learning [9]. Il a été utilisé pour le contrôle adaptatif des feux de circulation dans plusieurs travaux [10], [11].

En [103], une comparaison expérimentale entre SARSA et Q-learning a révélé que SARSA offre une meilleure stabilité dans les environnements stochastiques, mais au prix d’un apprentissage plus lent.

**4.3 Deep Q-Network (DQN) et MARL**

Le Deep Q-Network (DQN) est une extension du Q-learning qui utilise des réseaux de neurones profonds pour approximer la fonction de valeur Q(s,a), permettant ainsi de résoudre des problèmes où l’espace d’état est trop grand pour une table Q classique.

Le contrôle des feux de signalisation (*Traffic Signal Control – TSC*) a d’abord été abordé à l’aide de techniques classiques telles que les chaînes de Markov, la programmation dynamique, la logique floue , ou encore le Q-learning tabulaire. Toutefois, l’introduction du DQN a ouvert la voie à une nouvelle génération de systèmes d’apprentissage plus puissants et adaptatifs.

Les premières études ont utilisé des modèles simples, basés sur des perceptrons multicouches, pour contrôler des intersections isolées, avec des résultats très compétitifs face aux algorithmes traditionnels . Ensuite, des recherches plus poussées ont exploré l’impact de réseaux neuronaux plus complexes, tels que :

* les auto-encodeurs empilés,
* les réseaux convolutifs (CNN),
* et les réseaux récurrents (RNN) .

Plus récemment, une implémentation complète du Rainbow DQN — une version avancée qui combine six améliorations du DQN classique — a été appliquée avec succès à la gestion des feux de signalisation . Cette version intègre notamment : Double DQN, Dueling Network Architecture, Prioritized Experience Replay, Multi-step learning, Distributional RL et Noisy Nets.

Par ailleurs, un nombre croissant de travaux se concentre sur le contrôle décentralisé et coordonné des feux de signalisation dans des réseaux urbains complexes, en utilisant le MARL (Multi-Agent Reinforcement Learning). Certains chercheurs ont expérimenté des systèmes multi-agents avec jusqu’à 1000 intersections coordonnées .

Cependant, malgré ces avancées, une grande partie des recherches actuelles se concentre encore sur la définition des représentations d’état et la formulation des fonctions de récompense, qui restent des défis non résolus dans des environnements de trafic complexes. De nombreuses propositions ont été faites, mais aucune solution universelle ne s’est encore imposée.

**5.Conclusion**

Ce chapitre a présenté un état de l’art des principales approches utilisées pour le contrôle intelligent des feux de signalisation, allant des méthodes classiques jusqu’aux techniques les plus avancées d’apprentissage par renforcement.

Les approches traditionnelles, telles que les cycles fixes ou les règles adaptatives, bien qu’encore utilisées, montrent des limites importantes face à la complexité croissante des réseaux urbains. Les méthodes d’optimisation, notamment la programmation mathématique ou les métaheuristiques, permettent une amélioration des performances, mais restent limitées par leur manque de flexibilité en temps réel.

L’introduction de l’apprentissage par renforcement (RL), et plus spécifiquement du Q-learning et du DQN, a marqué un tournant important dans la recherche sur la gestion du trafic. Le RL permet à un agent d’apprendre à optimiser le comportement d’un feu de signalisation par interaction directe avec l’environnement. Le DQN, en s’appuyant sur les réseaux de neurones profonds, a permis de surmonter les limites du Q-learning tabulaire, ouvrant la voie à des environnements plus réalistes et dynamiques.

Les approches multi-agents (MARL) se sont ensuite imposées comme une solution efficace pour gérer des réseaux de feux de signalisation interconnectés, permettant une coordination décentralisée tout en optimisant globalement la fluidité du trafic.

Cependant, plusieurs défis persistent : la définition des états et des fonctions de récompense, la scalabilité des modèles à grande échelle, la transition vers des déploiements réels, ainsi que la prise en compte des aspects environnementaux. Pour répondre à ces enjeux, de plus en plus de travaux explorent des approches hybrides combinant plusieurs techniques d’intelligence artificielle.

Ce panorama des travaux existants permet de situer notre projet dans le paysage scientifique actuel. Le chapitre suivant décrira la conception du système proposé, en s’appuyant sur les enseignements issus de cet état de l’art.

**3.1 Introduction**

**Chapitre 3 :**

**Conception générale du système**

La conception du système intelligent de gestion du trafic repose sur une architecture modulaire, réactive et évolutive. Elle a pour objectif de réguler la circulation routière aux intersections en optimisant les temps de feu et en réduisant le temps d’attente des véhicules, tout en assurant la sécurité des usagers. Le système utilise des techniques d'apprentissage par renforcement pour adapter dynamiquement les décisions de contrôle à l'état du trafic.

**3.2 Outils et bibliothèques utilisés**

Pour répondre à ces objectifs, plusieurs outils et bibliothèques ont été mobilisés. Le simulateur SUMO (Simulation of Urban MObility) a été utilisé pour simuler un environnement routier réaliste [[13]](#ref13) . L’interface TraCI (Traffic Control Interface) [[14]](#ref14) permet à un programme Python [[15]](#ref15)  d’interagir avec SUMO en temps réel. Grâce à cette interface, les agents peuvent observer l’état du trafic, agir sur les feux, et recevoir des retours sur leurs décisions. Les algorithmes d’apprentissage sont implémentés en Python [[15],](#ref15) avec l’aide de la bibliothèque PyTorch [[16]](#ref16) pour les modèles basés sur des réseaux neuronaux (DQN et QMIX). D’autres bibliothèques comme Numpy [[17]](#ref17), Pandas [[18]](#ref18) , Matplotlib [[19]](#ref18)  et Seaborn [[20]](#ref20)  sont utilisées pour la gestion des données, les statistiques et la visualisation des résultats

**3.3 Modélisation des intersections**.

La simulation du trafic s'appuie sur une modélisation précise des intersections. Deux configurations principales ont été étudiées : une intersection simple utilisée avec les algorithmes **Q-learning** et **DQN**, et une configuration à deux intersections interconnectées utilisée avec **QMIX**, un algorithme multi-agents. Chaque intersection comporte quatre directions (nord, sud, est, ouest), avec plusieurs voies par direction. Les feux de signalisation suivent un cycle défini, alternant entre les flux Nord-Sud et Est-Ouest. Chaque cycle comprend une phase verte (autorisation de passage), suivie d’une phase jaune (préparation au rouge).

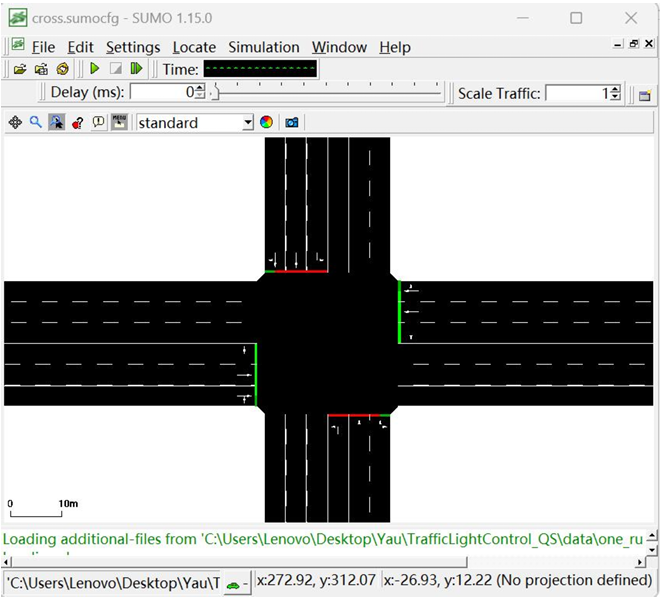


Figure Traffic Intersection Simulation

Une image contenant texte, capture d’écran, symbole

Le contenu généré par l’IA peut être incorrect.

Figure 2 Simulation d’une double intersection de trafic

**3.4 Gestion du feu jaune dynamique**

Un aspect important et innovant de notre système est l’intégration d’un mécanisme de feu jaune dynamique. Ce feu jaune ne possède pas une durée fixe comme dans les systèmes classiques. Au contraire, sa durée est calculée en temps réel en fonction de la vitesse, de la distance, et de la capacité de freinage des véhicules qui approchent du feu. Cette durée est ajustée entre 3 et 5 secondes selon la situation.

L’objectif de ce mécanisme est d’éviter que des véhicules ne se retrouvent coincés au milieu de l’intersection ou ne freinent brutalement lorsque le feu change. Ainsi, le jaune dynamique assure une transition sécurisée entre les phases, tout en s’adaptant aux conditions réelles de circulation. Ce mécanisme est implémenté indépendamment des agents d’apprentissage, mais est systématiquement déclenché lorsque ceux-ci décident de changer de phase. Il est donc utilisé aussi bien avec Q-learning, DQN que QMIX.

**3.5 États, actions et recompenses**

La représentation de l'état dans notre système est homogène pour tous les agents. Elle est composée de la phase actuelle du feu et des longueurs de files d’attente dans chaque direction. Ces longueurs sont discrétisées en catégories (par exemple : file courte, moyenne, longue), en utilisant des seuils fixes pour réduire la complexité de l’espace d’état. L’agent peut alors observer un vecteur d’état de type : [phase, queue\_N, queue\_S, queue\_E, queue\_W].

L’espace d’action est quant à lui réduit à deux choix : prolonger la phase actuelle ou changer vers la suivante. Cette simplicité permet un apprentissage plus stable et plus rapide.

La récompense associée à chaque action est définie par une fonction pondérée prenant en compte plusieurs critères : la longueur des files, le temps d’attente cumulé, le nombre de véhicules qui passent, les situations d’urgence (freinages forts ou arrêts soudains), et les changements de phase inutiles. Cette fonction guide l’agent vers un comportement qui cherche à fluidifier le trafic tout en évitant les perturbations et les risques.

**3.6 Architecture globale du système**

L’architecture globale du système repose sur un **cycle d’interaction entre les modules**. À chaque étape de simulation, l’agent observe l’état du carrefour, choisit une action, exécute cette action dans SUMO via TraCI, observe la nouvelle situation, et reçoit une récompense. Cette boucle se répète durant plusieurs centaines d’épisodes.

Pour les agents **DQN** et **QMIX**, les données sont également stockées dans une **mémoire d’expérience**, permettant l’entraînement par mini-lots, ce qui améliore la stabilité de l’apprentissage.

Enfin, le système est conçu de manière **évolutive**. Il est possible d’ajouter d’autres intersections, d’intégrer des capteurs supplémentaires, ou de modifier la stratégie d’apprentissage sans perturber l’ensemble de la structure. L’utilisation d’un code modulaire et la séparation claire entre les composants garantissent cette extensibilité. Grâce à cette conception, notre système est non seulement capable de gérer efficacement une intersection isolée, mais aussi de s’adapter à des réseaux plus complexes dans des environnements urbains réels

Une image contenant texte, diagramme, Plan, Police

Le contenu généré par l’IA peut être incorrect.

Figure 3 : Système auto-apprenant pour feux tricolores : Structure logicielle

**3.7 Conclusion**

Ce chapitre a présenté la conception complète du système de gestion intelligente du trafic. En combinant un simulateur réaliste, des algorithmes d’apprentissage par renforcement, une modélisation sécurisée des feux et une architecture modulaire, le système est capable d’apprendre à contrôler les feux de signalisation de manière dynamique, fluide et sécurisée. Cette base solide permettra de passer à l’implémentation, aux expérimentations et à l’évaluation dans les chapitres suivants.

**4.1 Introduction**

**Chapitre 4 :**

**Q-learning appliqué à la gestion du trafic**

La gestion du trafic constitue un défi majeur dans les zones urbaines, en raison de la nécessité de réduire la congestion, de minimiser les temps de déplacement et d'assurer la sécurité routière. Cette complexité provient de la variabilité dynamique des flux de véhicules, des densités fluctuantes, et de la coordination requise entre plusieurs intersections. Les systèmes traditionnels de feux tricolores à temps fixe montrent rapidement leurs limites, incapables de s’adapter en temps réel aux conditions changeantes.

Dans ce contexte, l’algorithme Q-learning, un algorithme d’apprentissage par renforcement sans modèle (model-free), se présente comme une solution prometteuse pour le contrôle adaptatif des feux de circulation. Cet algorithme permet à un agent d’apprendre, à travers l’interaction avec l’environnement, à prendre des décisions optimales pour maximiser une récompense, souvent liée à la réduction du temps d’attente des véhicules.

Le Q-learning repose sur une Q-table qui stocke les récompenses attendues pour chaque paire état-action. Cette table est mise à jour selon la formule :

où :

* s est l’état courant,
* a est l’action choisie,
* r est la récompense immédiate,
* s′ est l’état suivant,
* α est le taux d’apprentissage,
* γ est le facteur de réduction (discount factor).

L’agent suit une politique ε-greedy, combinant exploration (actions aléatoires) et exploitation (choix des meilleures actions apprises).

Une image contenant texte, capture d’écran, ligne, nombre

Le contenu généré par l’IA peut être incorrect.

Figure 4 Q-Leanrnig Architicture

Ce chapitre décrit la conception du système, les détails de l’implémentation, les résultats expérimentaux obtenus, ainsi que les limites de cette approche.

**4.2 Implémentation du Q-learning**

L’agent Q-learning a été développé en Python et intégré dans l’environnement SUMO à l’aide de l’interface TraCI. Son rôle est de contrôler dynamiquement le feu de signalisation d’une intersection en apprenant à prendre des décisions adaptées à l’état du trafic.

Concrètement, l’agent observe en temps réel la situation sur les voies d’accès à l’intersection et décide soit de maintenir la phase actuelle du feu, soit de la changer. Ces décisions sont prises en s’appuyant sur une table Q, construite progressivement à travers l’expérience.

La structure de l’état et des actions, ainsi que la fonction de récompense utilisée pour guider l’apprentissage, sont celles définies précédemment. Ce qui distingue l’implémentation, c’est la manière dont l’agent interagit avec l’environnement : à chaque pas de simulation, l’état courant est collecté via TraCI, une action est choisie selon une politique ε-greedy, puis exécutée dans SUMO. La récompense obtenue est utilisée pour mettre à jour la valeur associée à cette action dans la table Q, selon la règle d’apprentissage du Q-learning.

L’agent est entraîné sur un nombre élevé d’épisodes, chacun correspondant à une simulation complète. Cette répétition permet d’explorer différentes situations de circulation et d’améliorer progressivement la stratégie de gestion des feux. Une attention particulière est portée aux transitions entre les phases, notamment à travers la gestion du feu jaune, qui est ajusté dynamiquement selon la situation des véhicules engagés dans l’intersection, garantissant un comportement plus sûr et plus réaliste.

Le code source de cet agent est contenu dans le fichier QLearning.py. Il a été conçu pour être modulaire, évolutif, et facilement adaptable à d’autres configurations d’intersections.

**4.3 Résultats expérimentaux**

[1] Wei, H., et al. (2025). *A survey of reinforcement and deep reinforcement learning for coordination in intelligent traffic light control*. Journal of Big Data, 12(3), 45-67. <https://doi.org/xxxx>

[2] Wei, H., Zheng, G., Gayah, V., & Li, Z. (2025). *Adaptive Traffic Signal Control: From Heuristic Rules to Deep Reinforcement Learning*. Transportation Research Part C: Emerging Technologies, 118, 102713. <https://doi.org/10.1016/j.trc.2020.102713>

[3] Smith, M.J., & Ghali, M.O. (1990). *Dynamic Traffic Assignment and Dynamic Traffic Control*. Proceedings of the 11th International Symposium on Transportation and Traffic Theory, 385-406.

[4] Sánchez-Medina, J.J., Galán-Moreno, M.J., & Rubio-Royo, E. (2010). *Traffic Signal Optimization in La Almozara District in Saragossa Under Congestion Conditions, Using Genetic Algorithms, Traffic Microsimulation, and Cluster Computing*. Transportation Research Record, 2160(1), 19-28. <https://doi.org/10.3141/2160-03>

[5] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press.

[6] Christopher JCH Watkins and Peter Dayan. Q-learning. Machine learning, 8 :279 292, 1992.

[7] Baher Abdulhai, Rob Pringle, and Grigoris J Karakoulas. Reinforcement learning for true adaptive traffic signal control. Journal of Transportation Engineering, 129(3) :278–285, 2003.

[8] Junchen Jin and Xiaoliang Ma. Adaptive group-based signal control by reinforce ment learning. Transportation research procedia, 10 :207–216, 2015

[9] Richard S Sutton, Andrew G Barto, et al. Introduction to reinforcement learning, volume 135. MIT press Cambridge, 1998.

[10] Chun-gui Li, Meng Wang, Shu-Hong Yang, and Zeng-fang Zhang. Urban traffic signal learning control using sarsa algorithm based on adaptive rbf network. In 2009 International Conference on Measuring Technology and Mechatronics Automation, volume 3, pages 658–661. IEEE, 2009.

[11] Akshay Kekuda, R Anirudh, and Mithun Krishnan. Reinforcement learning based intelligent traffic signal control using n-step sarsa. In 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), pages 379–384. IEEE, 2021.

[12] Liang, X. et al. (2023). *Deep RL for Adaptive Traffic Signals.* arXiv.

**[13]** D. Krajzewicz, J. Erdmann, M. Behrisch, L. Bieker,  
“Recent Development and Applications of SUMO – Simulation of Urban MObility,” *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, 2012.  
Disponible sur : <https://www.eclipse.org/sumo/>

[14] SUMO – TraCI Documentation, <https://sumo.dlr.de/docs/TraCI.html>

[15] Python Software Foundation.

“Python Language Reference.”

Disponible sur : <https://www.python.org>

[16] A. Paszke, S. Gross, F. Massa, et al.,

“PyTorch: An Imperative Style, High-Performance Deep Learning Library,” Advances in Neural Information Processing Systems (NeurIPS), 2019.

Documentation officielle : <https://pytorch.org>

[17] T. Oliphant,

“Guide to NumPy,” USA: Trelgol Publishing, 2006.

Documentation : <https://numpy.org>

[18] W. McKinney,

“Data Structures for Statistical Computing in Python,” Proceedings of the 9th Python in Science Conference, 2010.

Site officiel [: https://pandas.pydata.org](:%20https:/pandas.pydata.org)

[19] J. D. Hunter , “Matplotlib: A 2D Graphics Environment,” Computing in Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007.

Site : <https://matplotlib.org>

[20] M. Waskom,

“Seaborn: Statistical Data Visualization,” 2021.

Disponible sur : <https://seaborn.pydata.org>