Can AI reduce pollution? Tackling traffic congestion in Milan with smart stoplights

Roberto Bernardelli, Emanuele Coccia, Domitilla Izzo
20599 Simulation and Modeling
2022/2023 - Group 12

ABSTRACT

Air pollution in Milan has reached hazardous levels in 2022 and 2023, on a trend with no sign of decreasing. The geographical location of Milan, in the middle of the Po valley, prevents wind circulation and enhances the negative effects of human activities, such as traffic from urban mobility. In an effort to tackle this problem, we propose the adoption of smart stoplights to optimize the traffic flow of the city and reduce waiting time at intersections. In the following paper, we first build an Agent-Based Model to simulate the road intersection of Viale Bligny and Via Bocconi; subsequently, we calibrate the agent attributes using real traffic data; and lastly, we train the stoplights with Reinforcement Learning to optimize their switching tempo. Our results show a decrease of 33% in waiting time for cars; this must not be intended as a generalized result but as an early result that gives our research promising prospects. In fact, the model we utilize has been built as a demo, and it is meant to be expanded with better data and on a larger scale in the future.

I INTRODUCTION

"The world's leading cities recognize that communities need clean air for their health and livelihoods", Antha Williams, the environment program director at Bloomberg Philanthropies, said.

Despite the fact that it has imposed limitations and implemented measures to enhance air quality, Milan has one of the worst air pollution levels in Europe and worldwide. The Italian city was reported as the third most polluted city in the world on Tuesday, March 21, 2023, behind Tehran and Beijing. On that day, the average PM10 level in Milan was 47.75 g/m^3, and there had already been 27 days since the start of the year when the "red alert" level of 50 g/m^3 was exceeded [1].

According to the air quality index (AQI) monitoring site map, which conveys the air quality values published and validated by the Regional Agencies for Environmental Protection (ARPA), despite an annual reduction in highly polluting cars and total emissions, heavy traffic and congestion in large metropolitan areas continue to be a major source of pollution [2]. Indeed, vehicle transport is said to account for 16% of total PM2 emissions, 5,66% of nitrogen oxide (NO) emissions, 18% of sulfur oxides (SO), and 20% of NMVOCs (mostly benzene). The sharp rise in the concentrations of microscopic particulate matter PM2.5, which in Italy has gone from an average of 19.3 g/m^3 to 19.4 g/m^3 in recent years, is especially scaring because PM2.5 is the cause of the majority of breathing and heart problems and is responsible for over 9% of deaths among Italians over the age of 30 [2].

Moreover, since 2019 (the last pre-Covid year), the transport sector has been responsible for roughly 25% of all greenhouse gas emissions and 30% of total CO2 emissions in Italy, with road transport accounting for approximately 93% of these. While overall emissions in Italy decreased by 19% between 1990 and 2019, transportation was one of the few sectors to experience an increase (+3.2% compared to 1990) [3]. Reducing CO2 emissions from transportation is thus critical for meeting the European Union's emission reduction targets (-55% of emissions in 2030 compared to 1990).

Transportation management should rely on in-

telligent traffic solutions to reduce and prevent traffic-related pollution. Testing traffic management methods in the real world or test fields without first doing theoretical evaluations would be extremely time-consuming, expensive, and sometimes dangerous due to the complexity of controlling mobility in metropolitan settings. As a result, theoretical approaches for assessing the advantages of traffic and transportation management initiatives are required, and simulation frameworks have proven to be a valuable ally.

An agent-based approach was employed for the simulation, as it is generally well-suited to modeling complex traffic and transportation systems. The environment comprises the single Viale Bligny - Via Bocconi intersection in Milan, but the model developed seeks to be general and scalable to a network of intersections mapping entire neighborhoods or small cities, although with some necessary adjustments.

Traditional traffic signal control approaches, such as pre-timed control [4], actuated control [5], adaptive control [6], and optimization-based control, rely heavily on the given traffic model or predefined rules, and thus fail to adjust to dynamic traffic nicely. As a result, experts have begun to investigate reinforcement learning (RL) techniques for improvements in traffic signal regulation, and several studies have already shown that the use of RL outperforms traditional transportation control methods [7] [8]. The most important strength of RL is that it immediately learns how to execute subsequent actions by examining the feedback received from the environment after previous actions.

Among the leading RL techniques, OpenAI's state-of-the-art Proximal Policy Optimization (PPO) algorithm was chosen to be applied on top of the simulated environment in an attempt to optimize traffic light time switching. In fact, PPO is the ideal option because it shares the data efficiency and consistent performance of earlier RL techniques while being simpler to implement, more versatile, and more robust.

The combination of a tailored ABM model for simulation and the efficiency of the PPO algorithm is intended to provide the foundation for innovative solutions to traffic-related pollution in Milan.

The study begins with an overview of previous studies relevant to the topic at hand, followed by a description of the proposed ABM model for simulation, as well as how it was conceptualized and developed. The data collection and process for calibrating the model parameters are then presented. Finally, the application of the PPO for RL on traffic light switching time is discussed, as are the expected improvements in average waiting time resulting from the simulation.

II LITERATURE REVIEW

Vehicle emissions are the primary source of roadside nitrogen oxides (NOx) in the world [9], which, along with particulate matter, ground-level ozone (O3), and ammonia (NH3), are major air pollutants with substantial health and environmental consequences [10] [11]. Additionally, the atmosphere's concentration of greenhouse gases like CO_2 is constantly rising [12]. Huang et al. [13] pointed out that both fuel energy consumption and greenhouse gas emissions caused by the transportation sector will exceed 50% by 2030. Like the rest of the world, the EU experiences large traffic-related increases in greenhouse gas concentrations and other air pollutants, which cause pollution levels in many European cities to exceed EU guidelines [14].

Therefore, improving traffic conditions is one approach to reducing emissions. Cars may travel farther and produce fewer harmful pollutants if they stop less frequently, all while using less fuel and improving road safety [15]. Using the Smart Emissions Measurement System (SEMS) created by TNO, Deschle et al. calculated the total quantity of CO_2 and NOx emitted by vehicles at a specific distance around various crossings [16]. The cars were grouped according to whether they slowed down, stopped, or did not halt at an intersection. The study showed that the cluster of cars that stopped differed significantly from the other two. According to their estimates, avoiding a stop may cut CO_2 emissions at each intersection by 0.32 kg and NOx emissions by 75% over a 2-kilometer distance.

Several estimation models attempted to quantify journey time, energy consumption, and greenhouse gas emissions depending on vehicle speed and traffic circumstances. Chen et al. [17] reported that frequent acceleration and deceleration at low speeds, particularly in congested regions, are the major factors that aggravate vehicle emissions. Li et al. [18] investigated signal timing

models to improve traffic quality while lowering fuel consumption and pollutants at intersections, using the average delay per vehicle at an intersection as a traffic quality indicator. As a result of these investigations, stop-and-go traffic due to congestion at signalized intersections seems to increase fuel consumption and emissions.

Kamran et al. [19] emphasize the importance of traffic light timing in decreasing congestion at major urban junctions. To optimize the timing, they propose a simulation model aimed at minimizing the average time that vehicles spend in the system using real data, such as vehicle arrival and departure times and time spent waiting at a red light, collected by monitoring a specific intersection.

Therefore, reducing the average time a car spends at a junction appears to be a suitable strategy for reducing congestion and, in turn, cutting emissions and improving air quality.

An agent-based approach is generally suited to the domain of traffic and transportation systems [20]. Indeed, Chen and Cheng [21] provide an overview of how techniques and methods developed in the agent system and multi-agent system (MAS) fields have been applied to many aspects of traffic and transportation systems, including modeling and simulation, intelligent urban traffic control (UTC), congestion management, driver-infrastructure collaboration, and decision support.

Zhao et al. [22] developed an agent-based macroscopic traffic simulation model for the city of San Francisco, with the goal of striking a balance between abstractions and specifics to provide efficient city-scale analysis. During the one-week time span examined in the study, the hourly travel demand added up to roughly 9 million trips for nearly 9 million agents, and a priority-queue-based Dijkstra algorithm was used to speed up the shortest path computation for each agent. Overall, the ABM model displayed strong computational performance as well as the capacity to reasonably simulate city-level traffic.

However, due to the increased detail added in tracking each car individually, microscopic simulations can more realistically represent traffic flow than macroscopic simulations and are more widely used to perform traffic operations analysis and evaluate the merits of new traffic control and management technologies [23].

Microscopic models have proven useful to mimic urban traffic processes both for individual intersections or networks of urban intersections. The level of detail enables simulation of the vehicle-following process as well as the development of ways of resolving vehicle conflicts. The standards of behavior are based on valid traffic rules and regulations that stipulate things like maximum speed and right of way, and they often include methods for managing vehicle speed and acceleration [24].

Microscopic traffic flow models are classified into three types: acceleration (car-following) models for longitudinal movements, lane-changing models for lateral movements, and discrete-choice models for specific on-route decisions [25].

Car-following models include safe-distance models, where vehicles alter their speed to maintain a safe distance from the vehicle in front [26]. The car-following model proposed by Pipes is one such model, and it is based on a suggestion found in the California Vehicle Code: 'A good rule for following another vehicle at a safe distance is to allow yourself the length of a car for every ten miles per hour you are traveling' [27]. In other models, the safe distance is estimated as the distance required to avoid a collision if the leader decelerates rapidly. The first model of this type was proposed by Kometani and Sasaki in 1959 [28] and improved by Gipps in 1981, where the vehicle is considered bound by a desirable speed in free flow conditions and a safe speed when following another vehicle [29].

Therefore, since the 1950s, several car-following models with diverse strategies have been developed. Despite the number of models that have already been built, there is ongoing active research in the field. As a result, it is reasonable to believe that the perfect car-following model has yet to be discovered or that no such thing as a "perfect model" exists. Every car-following has advantages and limitations, and the most appropriate one may vary according to the application [26].

Experts recognize Reinforcement Learning (RL) as a promising technology that may be easily used to test potential solutions to the problem of traffic light control in a simulation. The first advantage of utilizing RL to develop a signal controller is that it can learn and dynamically adapt to changes in the traffic environment. Second, while the process of reinforcement learning is driven by

a reward or a penalty, the RL algorithm does not need to develop an environment-specific mathematical model. Savithramma et al. [30] used RL with a reward designed for reducing delay and matching green time with traffic volume. Moreover, they coupled it with the Gradient Boosting Regression Tree (GBRT) to reduce state complexity. Given an initial traffic queue and a set vehicle speed during green time, as well as subsequent traffic flow fluctuations, the primary goal of their signal controller was to decide green timing for each phase based on the condition of traffic flow. The results of the study demonstrated a significant reduction in waiting delay and traffic queue size adopting the proposed control scheme based on RL.

Abdulhai et al. [31] used Q-learning to manage traffic lights at a single intersection. As the traffic state, they also chose the queue lengths on the approaching links, as well as the elapsed phase time, and defined the reward as the total delay of cars incurred between consecutive decision points. Under variable traffic conditions, test results suggest that the Q-learning controller outperforms the pretimed signal timing.

Deep Q-learning methods, as well as "vanilla" policy gradient methods and trust region natural policy gradient methods, are now among the best options for RL with neural function approximators.

However, in 2017, OpenAI proposed a new class of methods called Proximal Policy Optimization (PPO), which have the data efficiency and reliable performance of trust region policy optimization (TRPO), but are much simpler to implement, more general, and empirically have better sample complexity because they only use first-order-approximation [32]. PPO algorithm is defined by an objective function with clipped probability ratios, which forms a pessimistic estimate (i.e., lower bound) of the policy's performance. It works by conducting numerous epochs of minibatch updates, which makes the algorithm scalable to large models and parallel implementations as well as data efficient and robust (i.e., successful on a wide range of challenges without the need for hyperparameter tuning). Schulman et al. compare the performance of several variants of the surrogate objective function for PPO and show that the one with clipped probability ratios performs best [32]. They also compare PPO to numerous earlier

RL techniques from the literature, including the ones mentioned above, demonstrating how it outperforms all of them, particularly on continuous control.

This research intends to add to the literature by offering a new model of microscopic traffic simulation that, while developed for a single intersection, is scalable to a network of intersections. The combination of ABM modeling with cutting-edge RL techniques (i.e., PPO) aims to provide a starting point and insights for improving traffic-associated pollution in Milan.

III   AGENT-BASED MODEL

In this section, we present our approach to the road simulation. It is worth mentioning that the design of our code is, for the most part, original. As a matter of fact, the following discussion will seldom contain references since the uniqueness of the problem required custom-made solutions that could not be found elsewhere in the literature.

The main objective of our model is to represent the effect of stoplight switches on traffic flow at an intersection. The secondary objective is to build a model that can be potentially scaled from one intersection to an entire neighborhood or even a small city.

Initially, while sketching our project, we had to decide whether to choose a System Dynamics approach or an Agent-Based Model.

In a System Dynamics fashion, we would represent the car flow as a system of differential equations: a red stoplight would increase the stock of cars waiting before the intersection, while a green stoplight would decrease it. This is a simple approach that can work well on one or a few intersections; however, as the complexity of our model increases, it is hard to predict the behavior of the traffic flow with only differential equations since the flow at every intersection depends on the flow management of the others. For this reason, and because our secondary objective is to create a scalable model, we opted to build an Agent-Based Model.

An Agent-Based Model is better suited when modeling complex systems: by modeling tiny particles (cars) instead of a fluid (traffic flow), we can study phenomena that emerge when those particles interact with each other and that are too complex to model otherwise (e.g., how conges-

tion happens). Moreover, with ABM, we can assume heterogeneity between the agents; for example, we can introduce chaos in the model by assigning different driving styles to the car drivers. The main inspiration for the simulation was a basic model made by Professor Jan Kwakkel from the Technical University of Delft, whose demo[1] shows several cars moving in a circle. His model is built on a simple yet powerful concept: cars are accelerating when the distance from the next one is high, and decelerating when the distance is short. From here, we had all the instruments we needed to realize our complex simulation.

The system was built in Python, but without using the Mesa package, a popular library for ABM, since we thought building our own classes would give us more freedom and expressiveness. Moreover, we simulated only one intersection, but we believe that with more time and resources, it would be easy to map an entire neighborhood or even a small city, as our code is easy to scale.

Fig. 1: Mapping of the Viale Bligny - Via Bocconi intersection
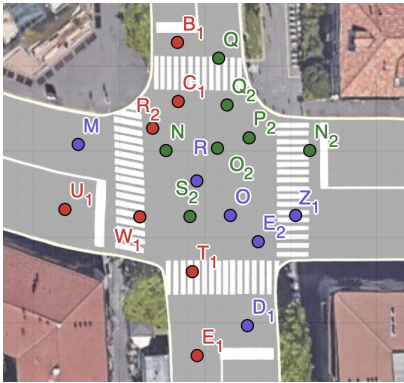
The first component of our model is the environment, which is where cars live. Taking inspiration from Zhao et al. [22], we decided that the best idea was to map the intersection as a directed graph, where each path is a series of connected nodes. Nodes are the best way to represent roads because we can approximate curves and turns easily. The mapping of the whole intersection turned out to be the one in Figure 1.

All the nodes live in a Cartesian coordinate system, and they belong to the same parent class "Node". The most important child classes are *Stoplight*, *ListNode*, and *YieldNode*.

[1]https://github.com/quaquel/EPA1324_open/blob/master/src/week_2/cars_on_circle.ipynb

The *Stoplight* class is used for a node that represents a stoplight, and it has a binary state that corresponds to red light or green light. The *ListNode* class contains a linked list where cars are assigned, and it helps regulate traffic: cars don't leave the linked list until they encounter another *ListNode*. The *YieldNode* class has the purpose of stopping cars if they need to give the right of way: it signals if cars are coming from the left, and in that case, it stops the traffic.

Nodes have the main purpose of giving direction to the cars, which keep track of the node in front of them. When cars are created, they are given a direction based on the normalized vector difference between the position of the next node and the position of the car - remember that we can imagine positions as vectors in a Cartesian system - and the direction is updated according to the next node whenever the current node of reference has been surpassed. To make the cars understand that they passed over a node, we check if the angle between their direction and the direction from the node of reference is greater than 45 degrees (an arbitrary threshold). A car will have an angle of 0 degrees until it surpasses a node, at which point the two directions will form an angle of 180 degrees, and the car will change the node of reference and update its direction.

Another important component of the environment are the aforementioned linked lists, which also help regulate traffic. Figure 2 depicts a schematic representation of what a piece (i.e., one-forth) of the environment looks like.

Now let us move the focus to the cars, which represent our agents. Cars belong to the *Car* class and have several attributes, among which four are used to model their behavior: initial speed ("speed" for short), maximum speed, acceleration, and safety distance (in the code, it is referred to as "lookahead").

The simulation is running at 32 fps, and at each step, which corresponds to a frame, the position of the cars is updated by calling the method "step" on the *Car* objects.

Cars can only look at the node in front of them and at the next car in their linked list (if any), and they regulate their speed depending on the distance from the object they have in front. Specifically, they enter the system with a speed equal to the initial speed, decelerate by cutting their speed in half at every frame if their distance from the
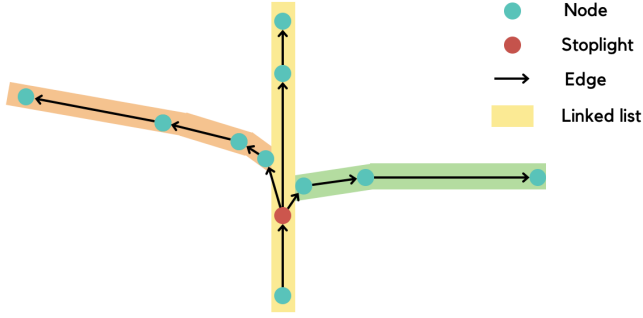
Fig. 2: Portion of the environment relative to cars coming from the south

next object is less than their safety distance (which is equal to a constant multiplied by the speed of the car), and accelerate if there is no object in front of them until they reach their maximum speed. The speed and the acceleration are random variables sampled from a normal distribution, with a mean equal to the values resulting from the calibration and a standard deviation of 1/20. This randomness was introduced to simulate different driving behaviors and make cars less predictable for our AI stoplight later on.

At the beginning of each iteration, cars are sampled from a Poisson distribution we estimated from 15 minutes of real-time traffic data. Each possible path (12 in our simulation) has its own Poisson distribution. While these conditional distributions are feasible to calculate for one intersection, they will not be when scaling this project in the future; for this reason, it will be advisable to use the methodology from Zhao et al. [22], where they sample the starting point and the final point independently – even if the independence assumption is hard to justify. Coming back to our simulation, when cars are spawned from the conditional Poisson densities, they get assigned a path calculated with the Dijkstra algorithm. This is done to ensure scalability because it would be cumbersome to hard-code every node sequence if we expanded our map. Cars are initially inserted into a linked list that helps them understand whether there is another car in front of them and whether they should slow down or not. When they turn right or left at the intersection, they also switch list.

When cars reach a node of the class *FinalNode*, which is a node with no additional properties other than being the final one, they are removed from

their linked list, and the step method returns the total time they have been living in the simulation. Stoplights could also be considered agents in the sense that they are able to switch from red to green and vice versa, and in this way they interact with the environment. A *Stoplight* can keep track of how many cars are in front of them because every node has an attribute called queue that keeps track of the cars that are just before them. The number of cars waiting in line will be an important component later on in the Reinforcement Learning algorithm.

IV   DATA COLLECTION AND CALIBRATION

The traffic at the Viale Bligny - Via Bocconi intersection was recorded for 15 minutes in three different perspectives, for a total of 45 minutes. The data collection focused on the south-north direction of Via Bocconi, which was then used for calibration.

The first video was shot from about 150 meters away, capturing how cars behaved as they approached the intersection from the south. The second captures the start of Via Bocconi in order to do the same for cars from north to south. This recording captured both the incoming cars and the complete intersection with the traffic lights. The final video showed cars in the intersection and traffic lights changing from the other side of the intersection.

Fig. 3: Recording Positions for Data Collection

The information necessary for the calibration was transcribed while examining the three videos together. For each car, the time in seconds it was spotted entering the system 150 meters from the intersection was documented, as well as the time

it was seen leaving the system near Largo Isabella d'Aragona. Data for cars that never arrived at the junction because they stopped or entered one of the side streets of Via Bocconi were discarded.

The number of vehicles entering the system from each direction, as well as whether they continued straight, turned right, or turned left, were then counted while looking at the entire intersection and both streets. This data enabled us to estimate the Poisson distribution from which the cars are sampled, as well as the chance of each car taking a specific path, allowing us to better adjust the simulation to what was happening in reality.

Alongside the data on the cars, the information on traffic light switches was annotated. The current time setting of traffic signals could be ascertained by tracing the seconds when they turned red or green. Given the model's simplification of only considering green and red lights, the yellow light was considered green because we observed that most people at the intersection tend to pass with the yellow light, despite the driving code saying otherwise, and it was thought reasonable to include the yellow light in the green one to avoid collisions if the model was applied to reality. According to the videos, the green light lasts around 30 seconds, whereas the red light lasts about 45 seconds. Furthermore, a 7-second delay was observed between the traffic signals on Via Bocconi turning red and the ones on Viale Bligny becoming green.

As previously stated, a single street (Via Bocconi) and a single direction (south-north) were considered for the calibration of the car parameters. As a consequence, the map used to calibrate the model differs from the one used for the simulation of the entire intersection in that it consists of a larger portion of Via Bocconi (such as to include the whole portion recorded) but a smaller portion of Viale Bligny (which is ignored for calibration purposes). Furthermore, it is important to note that when calibrating the model, we assumed no variation between cars or drivers; otherwise, using the grid search to determine the optimal car parameters (which are expected to be the same for each car) would have been impossible.

Multiple grid searches were conducted around parameter values that were deemed reasonable in light of the field observation and the simulated model. Because the Mean Absolute Error (MAE) has a low sensitivity to outliers, it was chosen as the loss function. Initially, the goal was to find the values of the four car parameters - maximum speed, speed, acceleration, and safety distance - that were closer to how cars behave in reality. However, since the loss in MAE appeared to be tied to a fall in the acceleration parameter until unbelievably small values, the best MAE was obtained for an acceleration level that was not credible and practical. This was most likely caused by the resource and data collection limitations, which will be soon covered in more detail.

As a consequence, the acceleration (measured as difference in velocities) was held constant to a reasonable level, and the grid search was focused on calibrating the other car parameters. The resulting parameters are associated with a higher MAE than before but appear to be considerably more indicative of reality. Due to the discrepancy in scaling between the two maps used for simulation and calibration, these parameters were scaled by a factor of 1.1.

Obviously, several simplifications have been made in this calibration approach. While the minutes recorded may be sufficient to calibrate the parameters that characterize car behavior, they are insufficient to accurately estimate the Poisson distribution from which cars are sampled. Indeed, a certain time of day, as well as very specific 15 minutes, were recorded and considered. If the proper instruments and resources were available, it would have been necessary to record the junction all day for at least a week to obtain a proper calibration. This would enable for different scenarios to be captured, taking into account variances in both the volume of vehicles at the intersection and the path they take at different times of the day and week (e.g., rush hours, weekdays, weekends).

V   OPTIMIZATION OF THE STOPLIGHTS

Once the model was calibrated, we needed to find the optimal way to configure the stoplights. Two main approaches were tried: one obtained through a grid search of the timings of red and green lights, and one using reinforcement learning.

A   Approach 1: Grid Search and Hard Coding of Stoplights

In our first approach, we employed a grid search technique combined with hard coding of stoplights. We varied the number of seconds required to make a switch between traffic signal phases and evaluated the waiting times for each configuration. For instance, a setting of 20 seconds meant that the stoplights switched every 20 seconds.

Through extensive experimentation using our ABM simulation, we conducted benchmarking on over 100 episodes, each lasting 3 minutes. The real-world configuration, which served as the baseline, exhibited an average waiting time of 25.92 seconds.
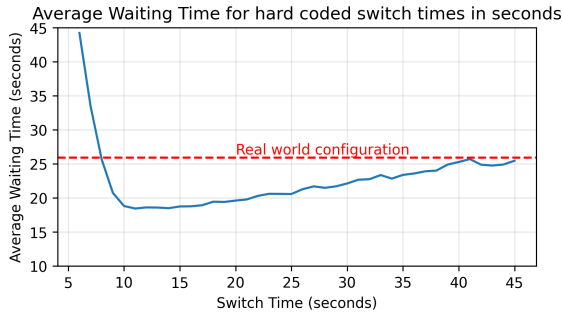
Fig. 4: Grid search results, in comparison with real world configuration

By employing the grid search approach, we were able to identify an optimal configuration that achieved a considerable reduction in waiting times. The configuration with the lowest average waiting time of 18.48 seconds outperformed the real-world setup, resulting in a notable improvement (28.7% improvement). It has been obtained by switching the stoplights every 12 seconds.

B   Approach 2: Reinforcement Learning

In our second approach, we employed the power of reinforcement learning (RL) to optimize the waiting times at the intersection. The RL agent was designed to control the four stoplights of the intersection and minimize the waiting times of vehicles passing through. To achieve this, we utilized the Stable Baselines 3 framework and OpenAI Gym to implement the RL environment.

We chose the Proximal Policy Optimization (PPO) algorithm as the RL model for training the agent. The PPO algorithm is well-suited for continuous control tasks and has proven to be effective in various RL applications. Here is a scheme of how the RL agent makes its decisions at each turn:
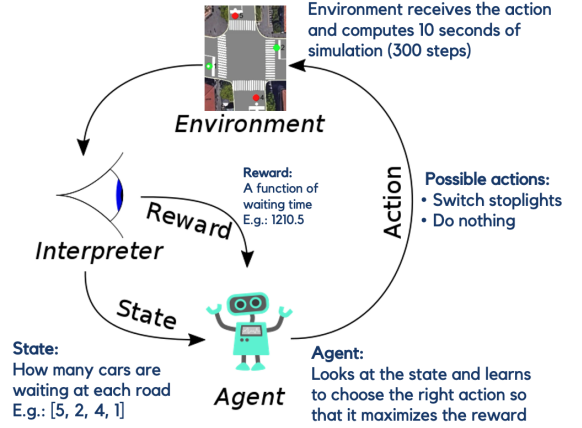
Fig. 5: Scheme of the Reinforcement Learning iterative training

The RL agent's decision-making process involves selecting one of two actions every 10 seconds of simulation: either switch the stoplights or do nothing. So, the action space is binary:

- Action 0: do nothing

- Action 1: switch the stoplights

The observation space provided to the RL agent includes the following information:

- The number of cars waiting to approach the intersection from the north-south and south-north directions.

- The number of cars waiting to approach the intersection from the east-west and west-east directions.

- A boolean value indicating the current direction of traffic flow, necessary so that the agent can have different behaviors based on the direction of the flow. It switches together with the stoplights.

- The number of steps since the last switch of the stoplights.

The reward function used in the RL training process is defined as follows:

$$reward = -(n\_waiting\_cars^2)$$

Here, n_waiting_cars represents the total number of waiting cars at the intersection, which includes vehicles approaching from the north, east, south, and west directions. The square of the waiting car count is multiplied by -1 to provide a reward that encourages minimizing the waiting times. We also tried to use the waiting times directly as the function to minimize, but the agent was having difficulty capturing the patterns of traffic, so we kept this proxy, which worked well.
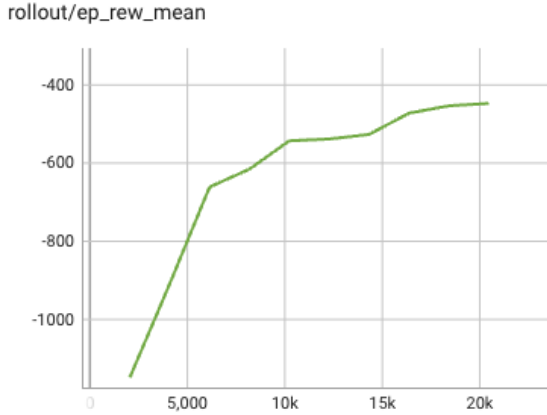

Fig. 6: Mean reward of the first 20k steps. The model quickly learns to solve the problem. Plotted with TensorBoard

We trained the reinforcement learning agent for 180 simulated days, in episodes of 3 minutes each. The model quickly learned to solve the problem in the first simulated week, but we continued training to simulate different random scenarios and improve stability.

To evaluate the performance of the RL agent, we conducted benchmarking experiments using our ABM simulation. Again, we performed 100 episodes, each lasting for 3 minutes. Remarkably, the RL agent achieved an average waiting time of 16.93 seconds, outperforming both the real-world configuration and the grid search optimization (34.7% improvement over the real-world configuration and 8.4% over the best grid search configuration)

The success of the RL agent can be attributed
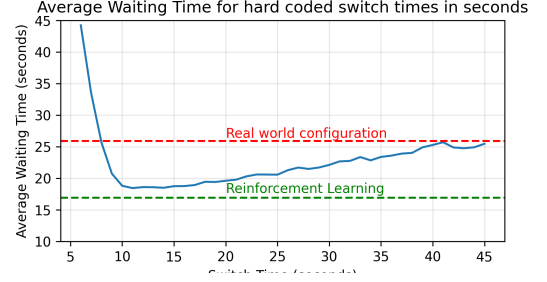
Fig. 7: Average Waiting Time in seconds for hard code switch times

to its ability to adapt the stoplight configuration based on real-time traffic conditions. By considering the counts of waiting cars from different directions and the history of switches, the RL agent learned to make informed decisions that minimize waiting times and optimize traffic flow at the intersection.

VI   REDUCTION IN POLLUTION

In this section, we attempt to give a rough estimate of the reduction in CO_2 emissions to quantify the impact of this potential traffic-control improvement.

According to a study [33] based on the city of London, cars emit on average 17 g of CO_2 per minute when idling. Since our model reduces waiting times by approximately 9 seconds (exact number: 8.99), we expect a reduction of 2.55 g of CO_2 per car. Assuming a flow of 1500 non-electric cars per day, we estimate a reduction of approximately 5 kg/day of CO_2 emitted for this intersection with the RL traffic control. By applying a similar algorithm to many more intersections, a significant amount of CO_2 emissions can be avoided.

It is not possible to give an estimate of the reduction in PM2.5. However, it is worth noting that minimizing waiting times and optimizing traffic flow can have indirect positive effects on air quality, including the reduction of PM2.5 pollutants.

VII   LIMITATIONS

Even though the model closely resembles real traffic, there are some important limitations that future works will need to address:

1. Pedestrians are not currently modeled. In our

simplified model, cars turn left and right without waiting for pedestrians to cross.

2. We are not taking into consideration reaction times. This aspect is currently embedded in the acceleration parameter, but to make the model even more realistic, it should be considered a parameter by itself.

3. More training data should be collected to get better parameters. Also, we only considered one intersection, and we probably overfitted the parameters for this scenario.

VIII CONCLUSION

In this study, we aimed to optimize waiting times at an intersection using two distinct approaches: a grid search combined with hard coding of stoplights and a reinforcement learning agent. Our benchmarking analysis revealed the following results:

1. The real-world configuration had an average waiting time of 25.92 seconds, serving as the baseline for comparison.

2. The grid search and hard coding approach led to an optimal stoplight configuration, achieving an average waiting time of 18.48 seconds. This approach demonstrated the potential for significant improvements over traditional setups.

3. The reinforcement learning agent outperformed both the real-world configuration and the grid search optimization, yielding an average waiting time of 16.93 seconds. The RL agent's ability to adapt to changing traffic conditions in real-time allowed for further reductions in waiting times.

These findings underline the importance of leveraging computational techniques to optimize traffic management. The grid search approach provides a systematic method for exploring various configurations, while reinforcement learning offers the flexibility to adapt to dynamic traffic conditions. Future research could focus on combining these approaches or exploring other optimization methods to further enhance traffic flow efficiency and reduce waiting times at intersections.
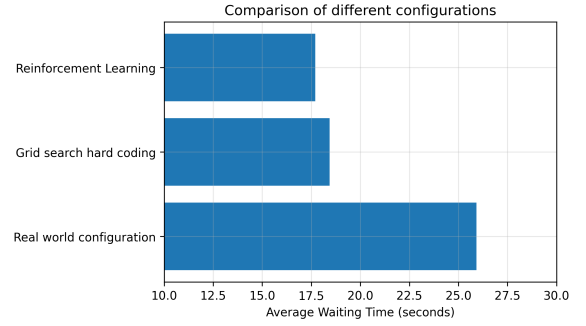
Fig. 8: Comparison of the real world configuration and the two optimization approaches

REFERENCES

[1] Ruetir. Milan, highly polluted air, is the third "dirtiest" city in the world, 2023. *https://www.ruetir.com/2023/03/milan-highly-polluted-air-is-the-third-dirtiest-city-in-the-world-news*. [Accessed: 2023-05-25].

[2] IQAir. Air quality index (AQI) and PM2.5 air pollution in Italy, 2023. *https://www.iqair.com/italy*. [Accessed: 2023-05-25].

[3] Italian Minister for Sustainable Infrastructure and Mobility. Decarbonising Transport: Scientific evidence and policy proposals. 2022.

[4] Peter Koonce and Lee A. Rodegerdts. Traffic signal timing manual. *Tech Report. United States. Federal Highway Administration*, 2008.

[5] Seung-Bae Cools, Carlos Gershenson, and Bart D'Hooghe. Self-organizing traffic lights: A realistic simulation. *Advances in Applied Self-Organizing Systems*, 2006.

[6] P.R. Lowrie, Roads, and Traffic Authority of New South Wales. Traffic Control Section. *SCATS, Sydney Co-Ordinated Adaptive Traffic System: A Traffic Responsive Method of Controlling Urban Traffic*. Roads and Traffic Authority NSW, Traffic Control Section, 1990.

[7] Elise van der Pol and Frans A. Oliehoek. Coordinated Deep Reinforcement Learners for Traffic Light Control. 2016.

[8] Samah El-Tantawy and Baher Abdulhai. Multi-Agent Reinforcement Learning for Integrated Network of Adaptive Traffic Signal Controllers (MARLIN-ATSC). *Conference Record - IEEE Conference on Intelligent Transportation Systems*, pages 319–326, 2012.

[9] E. E. McDuffie, S. J. Smith, P. O'Rourke, K. Tibrewal, C. Venkataraman, E. A. Marais, B. Zheng, M. Crippa, M. Brauer, and R. V. Martin. A global anthropogenic emission inventory of atmospheric pollutants from sector- and fuel-specific sources (1970–2017): an application of the Community Emissions Data System (CEDS). *Earth System Science Data*, 12(4):3413–3442, 2020.

[10] Thirupathi Boningari and Panagiotis G Smirniotis. Impact of nitrogen oxides on the environment and human health: Mn- based materials for the NOx abatement. *Current Opinion in Chemical Engineering*, 13:133–141, 2016. Energy and Environmental Engineering / Reaction engineering and catalysis.

[11] J E Jonson, J Borken-Kleefeld, D Simpson, A Nyíri, M Posch, and C Heyes. Impact of excess NOx emissions from diesel cars on air quality, public health and eutrophication in Europe. *Environmental Research Letters*, 12(9):094017, 2017.

[12] Carl Edward Rasmussen. Atmospheric carbon dioxide growth rate, 2023. *https://mlg.eng.cam.ac.uk/carl/words/carbon.html: :text=In%201960%2D1970%20the%20growth,about%202.27%20ppm%20per%20year*. [Accessed: 2023-05-07].

[13] Yuhan Huang, Elvin C.Y. Ng, John L. Zhou, Nic C. Surawski, Edward F.C. Chan, and Guang Hong. Eco-driving technology for sustainable road transport: A review. *Renewable and Sustainable Energy Reviews*, 93:596–609, 2018.

[14] European Environment Agency. Explaining road transport emissions, 2016. *https://www.eea.europa.eu/publications/explaining-road-transport-emissions/file*. [Accessed: 2023-05-06].

[15] Matthew J. Barth and Kanok Boriboonsomsin. Real-World Carbon Dioxide Impacts of Traffic Congestion. *Transportation Research Record*, 2058:163 – 171, 2008.

[16] Nicolás Deschle, Ernst Ark, Rene van Gijlswijk, and Robbert Janssen. Impact of Signalized Intersections on CO_2 and NO_x Emissions of Heavy Duty Vehicles. 2021.

[17] Changhong Chen, Cheng Huang, Qiguo Jing, Haikun Wang, Hansheng Pan, Li Li, Jing Zhao, Yi Dai, Haiying Huang, Lee Schipper, and David Streets. On-road emission characteristics of heavy-duty diesel vehicles in Shanghai. *Atmospheric Environment*, 41:5334–5344, 2007.

[18] Xiugang Li, Guoqiang Li, Su-Seng Pang, Xiaoguang Yang, and Jialin Tian. Signal timing of intersections using integrated optimization of traffic quality, emissions and fuel consumption: A note. *Transportation Research Part D: Transport and Environment*, 9:401–407, 2004.

[19] Mehdi A. Kamran, Hossein Ramezani, Sina Masoumzadeh, and Fatemeh Nikkhoo. Traffic Light Signal Timing Using Simulation. *Communications on Advanced Computational Science with Applications*, 2017:1–11, 2017.

[20] Fei-Yue Wang. Wang, F.Y.: Agent-based control for networked traffic management systems. IEEE Intell. Syst. 20(5), 92-96. *Intelligent Systems, IEEE*, 20:92 – 96, 2005.

[21] Bo Chen and Harry H. Cheng. A Review of the Applications of Agent Technology in Traffic and Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):485–497, 2010.

[22] Bingyu Zhao, K. Kumar, Gerard Casey, and Kenichi Soga. Agent-based model (ABM) for city-scale traffic simulation: A case study on San Francisco. pages 203–212, 2019.

[23] L.E. Owen, Yunlong Zhang, Lei Rao, and G. McHale. Traffic flow simulation using CORSIM. In *2000 Winter Simulation Conference Proceedings (Cat. No.00CH37165)*, volume 2, pages 1143–1147 vol.2, 2000.

[24] Thomas Schulze and Thomas Fliess. Urban Traffic Simulation With Psycho-Physical Vehicle-Following Models. 2000.

[25] Martin Treiber. *Traffic Flow Dynamics*. 2013.

[26] J. Olstam and A. Tapani. Comparison of car-following models. *Tech. rep. VTI - Swedish National Road and Transport Research Institute*, 2004.

[27] Louis Albert Pipes. An Operational Analysis of Traffic Dynamics. *Journal of Applied Physics*, 24:274–281, 1953.

[28] Eiji Kometani and Tsuna Sasaki. A Safety Index for Traffic with Linear Spacing. *Operations Research*, 7(6):704–720, 1959.

[29] P.G. Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111, 1981.

[30] Reinforcement learning based traffic signal controller with state reduction. *Journal of Engineering Research*, 11(1):100017, 2023.

[31] Baher Abdulhai, Rob Pringle, and Grigoris Karakoulas. Reinforcement Learning for True Adaptive Traffic Signal Control. *Journal of Transportation Engineering*, 129, 2003.

[32] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *ArXiv*, abs/1707.06347, 2017.

[33] Olivia Cairns Tim Barlow. Published project report PPR987. 2020.