sustainability

MDPI

Article

Traffic Signal Control via Reinforcement Learning for Reducing Global Vehicle Emission

Bálint Kővári [1], László Szőke [2], Tamás Bécsi [1,*], Szilárd Aradi [1] and Péter Gáspár [3]

[1] Department of Control for Transportation and Vehicle Systems, Budapest University of Technology and Economics, H-1111 Budapest, Hungary; kovari.balint@kjk.bme.hu (B.K.); aradi.szilard@kjk.bme.hu (S.A.)
[2] Robert Bosch Kft., H-1103 Budapest, Hungary; laszlo.szoke@hu.bosch.com
[3] System and Control Lab, Institute for Computer Science and Control, H-1111 Budapest, Hungary; gaspar.peter@sztaki.mta.hu
* Correspondence: becsi.tamas@kjk.bme.hu

Abstract: The traffic signal control problem is an extensively researched area providing different approaches, from classic methods to machine learning based ones. Different aspects can be considered to find an optima, from which this paper emphasises emission reduction. The core of our solution is a novel rewarding concept for deep reinforcement learning (DRL) which does not utilize any reward shaping, hence exposes new insights into the traffic signal control (TSC) problem. Despite the omission of the standard measures in the rewarding scheme, the proposed approach can outperform a modern actuated control method in classic performance measures such as waiting time and queue length. Moreover, the sustainability of the realized controls is also placed under investigation to evaluate their environmental impacts. Our results show that the proposed solution goes beyond the actuated control not just in the classic measures but in emission-related measures too.

Keywords: deep reinforcement learning; emission-reduction; sustainability; traffic signal control

Citation: Kővári, B.; Szőke, L.; Bécsi, T.; Aradi, S.; Gáspár, P. Traffic Signal Control via Reinforcement Learning for Reducing Global Vehicle Emission. *Sustainability* **2021**, *13*, 11254. https://doi.org/10.3390/su132011254

Academic Editor: Armando Cartenì

Received: 23 August 2021
Accepted: 7 October 2021
Published: 12 October 2021

1. Introduction

1.1. Motivation

According to the International Energy Agency (IEA), the transportation industry is responsible for 24% of the direct carbon dioxide emission, three-quarters of which came from road transportation in 2020. The reduction of CO_2 emission has evolved into a significant challenge because CO_2 is a massive contributor to climate change [1]. The emission of road transportation can be decreased in several ways, such as the utilization of less polluting fuels like hydrogen [2], more energy-efficient vehicles [3], or the application of intelligent transportation systems (ITS). ITS have the potential to mitigate congestion having a profound effect on several aspects of our lives. Such aspects are more productive hours, less emission, safer commute, and so forth. Consequently, this paper focuses on the more efficient operation of signalized intersections because they are the prime bottlenecks of urban transportation environments. Signalized intersections can be controlled more productively by replacing the fixed-time operation approaches with actuated control techniques simply by considering the incoming traffic flow of the crossings, which is called the Traffic Signal Control (TSC) problem.

1.2. Related Work

The TSC problem has been tackled with several different approaches over time. The earliest solutions utilized rule-based systems. These algorithms determine key states with unique actions that can be used to reach a performance criterion. The authors in [4] developed a Dynamic Traffic Signal Priority system that can handle real-time traffic and transit conditions. In [5] a new system has been developed called REALBAND that identifies platoons in the network then predicts their arrival time to the intersection with the

help of the fusion and filtering of traffic data. This paper also describes the solutions for the arrival of conflicting platoons. Another well-studied solution is Genetic Algorithms (GA). In [6] the authors developed a group-based optimization model that tackles the safety and efficiency issues of the stage-based signal control approach in the case of mixed traffic flows with unbalanced volumes. Ref. [7] proposes a novel Genetic Algorithm based approach that optimizes transit priority in scenarios with both transit and private traffic. Ref. [8] utilizes a simulation-based Generic Algorithm with a multi-objective optimization model that considers both delays and emission. Simulation-based solutions are also widely applied for the TSC problem. In [9] the authors have proposed a method that utilizes local communication between the sensors and the traffic lights and is capable of control them in a traffic responsive manner in complex real-world networks. Ref. [10] analyzed the behavior of the CRONOS algorithm and showed that it decreases the total delay compared to a local and a centralized control strategy. A significant number of researchers applied Dynamic Programming (DP) to solve the TSC problem, primarily because of its adjustability and potential to be used in a vast set of traffic conditions and its virtue to utilize a diverse group of performance indicators. In [11] the authors used an Approximate Dynamic Programming based approach that profoundly reduced the vehicle delays and also mitigated the required computational resources by approximating the value-function. Ref. [12] presents RHODES that decomposes and reorganizes the TSC problem into hierarchical sub-problems. The paper also showed that RHODES outperforms semi-actuated controllers in terms of delay. Multi-Agent Systems (MAS) can also be considered a common approach for tackling the TSC problem. Ref. [13] proposes a Group-Based signal control method where the signal groups are modeled as individual agents, and these agents can make decisions on the intersection level according to the given traffic scenario. Game Theory-based methods are well-explored in TSC applications. The authors in [14] formulated the TSC problem as a noncooperative game where the agents' goal is to minimize their queue length.

However, in recent trends, Reinforcement Learning (RL) algorithms have been used for solving the TSC problem. RL showed impressive results in this realm and also in several other transportation-related applications [15,16]. In [17] the authors proposed utilization of high-resolution event-based data for state representation and outperformed both fixed-time and actuated controllers in terms of queue length and total delay. Ref. [18] compared a Game Theory and RL-based solutions with fixed-time control for a single intersection set-up, and the results showed that both methods provide superior performance in queue length compared to the fixed-time controller. The authors in [19] proposed a dynamically changing discount factor for the Bellman equation that is responsible for generating the target values for the DQN algorithm. They compared their results with the fixed-time controller and the original DQN agent. The results showed that their improved agent reached better results in total delay and average throughput than any other solution. Ref. [20] compared conventional Reinforcement Learning algorithms with the DQN algorithm for the single-intersection TSC problem, and the results showed that the DQN algorithm mitigates more the total delay. Ref. [21] proposes a novel state representation approach called discrete traffic state encoding (DTSE). Compared to the same algorithm with feature vector-based state-representation, the new representation reaches better performance in average cumulative delay and average queue length. The authors in [22] developed an agent with hybrid action space that combines discrete and continuous action spaces and compared their solution to DRL algorithms with discrete and continuous action spaces and fixed-time control. The results showed that the proposed method outperformed all other methods in average travel time, queue length, and average waiting time. RL has also been applied for the multi-intersection TSC problem. Ref. [23] proposes a novel cooperative DRL framework called Coder that decomposes the original RL problem into sub-problems that have straightforward RL goals. The papers compare the proposed method to other RL approaches and also to rule-based methods such as fixed-time control. The authors in [24] proposed a new rewarding concept and the DQN algorithm, which is combined with a coordination algorithm for controlling multiple intersections. Ref. [25] proposed a

Multi-Agent A2C algorithm for solving the multi-intersection TSC problem and compared the results with other DRL algorithms both on synthetic traffic grid and real-word traffic network. For a more thorough review of algorithms that are applied for TSC, see [26] and for purely RL solutions for the TSC problem see [27,28].

These papers prove that RL can be successfully applied in this field. The main reasons for the utilization of RL in the TSC problem are the vast versatility of essential features that the RL framework possesses for sequential decision-making problems. One of them is the scalability from single intersections to multi intersections and even to complex networks. Others are the generalization feature of function approximators and the low computational cost of inferencing already trained models supporting real-life implementation. The mentioned advantages are the main reason why RL-based solutions have a vast literature in this domain. The trend of suitable algorithms in the TSC problem evolved along with the advances in RL. It started from a simple value-based and policy-based solution, such as Deep Q-Network (DQN) and Policy Gradient (PG), followed by the Actor-Critic approaches. The same happened in the case of state representations. In addition to feature-based value vectors, image-like representations started to trend thanks to the tremendous result of Convolution Neural Networks (CNN).

From the aspect of performance, one could consider the applied rewarding concepts as one of the most influential components of RL, but they have not matured at the same pace. The prevailing trends in the rewarding concept are mainly transportation-related measures, such as waiting time, queue length, vehicle discharge, phase-frequency, and pressure. Exploiting the vacancy, this paper aims to develop a novel rewarding concept that can outperform a modern actuated control approach in classic measures and, in the meantime, pays attention to the environmental aspect of the TSC problem.

1.3. Contributions of the Paper

The contribution of the paper is twofold. First, a novel rewarding concept is presented for the TSC problem, which avoids any reward shaping, hence being considered a new insight into this particular control problem. Second, a PG and a DQN agent are trained with this new rewarding concept to show how the performance formulates. Furthermore, our proposed solutions are compared against a delay-based actuated control algorithm. Moreover, the evaluation of this paper covers not just the classic measures, e.g., waiting time, travel time, and queue length, but also compares the different solutions in sustainability measures, such as CO_2, NO_x, CO emission, etc. Such evaluation can be argued as necessary due to the sustainable transportation, and the reduction of transportation emission have grown into major concerns all around the globe.

The paper is organized as follows: First, in Section 2 the utilized methodologies are detailed. Second, Section 3 the training environment along with its key components, such as the state representation, action space, and our novel rewarding concept, is presented. Last, Section 4 compares the performances of the different methods. In Section 5 the paper reveals a short conclusion about the results, discusses future endeavors, and ascertainment of the topic.

2. Methodology

2.1. Reinforcement Learning

The accelerated evolution of computer science and hardware resources enabled artificial intelligence and machine learning to have their renaissance. Deep Learning (DL) provides the opportunity of renewal to Reinforcement Learning (RL) as well. The recent results show that the combination of DL and RL can perform on a superhuman level in challenging domains [29,30]. These tremendous results fueled a lot of researches in a broad spectrum of applications such as robot control [31], autonomous driving [32], and in board games as well [33]. In 1998, Sutton and Barto revisited the concept of RL [34]. The building blocks and theory laid down by Sutton and Barto are used by hundreds of thousands of ar-

ticles and publications. Constant research tries to make RL more applicable and competent in solving real-life problems.

Reinforcement Learning can be interpreted as a method to tune artificial neural networks (ANN). Unlike Supervised Learning (SL), which is also a type of machine learning, RL does not use labeled training data as ground truth, which is hard or sometimes impossible to collect. RL generates its training data through interactions between the agent and the environment. At the same time, it tries to reach the optimal behavior through the maximization of a scalar feedback value called reward. The reward values implicitly expose the inner dynamics of the given control task since they quantify the effect of the executed action on the environment.

The RL formulation can be seen in Figure 1. The formulation consists of an environment and an acting agent. The interactions through action selection by the agent result in the state transition, and as feedback, rewards are given to the agent. In this particular domain, an environment can be a controllable simulator where the intersection is built. From the aspect of the TSC problem, a state is a collection of influential features that describe the intersection's dynamics. Actions that can change the environment's inner state are basically for manipulating the given traffic signal configuration, while the reward is created by considering traffic-related measures and its goal to operate as a performance objective that characterizes the quality of the agent's action The mathematical model of most RL algorithms is called Markov Decision Process (MDP). MDP is a mathematical framework for sequential decision-making. In an MDP, we suppose that the system has the Markov property, which means that the conditional probability distribution of the following state of a random process depends only on the current state, given all the previous states [34]. Thus the states should contain all information needed to draw this probability and make educated decisions based on the currently observable state.

An MDP can be defined with a 4 element tuple, $M \equiv < S, A, P_{S,A}, R >$, where

- S is the state space and $s_i \in S$ is the state at the ith time step,
- A is the action space and $a_i \in A$ means the action taken at the ith time step,
- $P_{s,a}$ is the probability of the state transition from state $s \in S$ with action $a \in A$. More generally it is denoted with: $p(s'|s,a)$.
- R stands for the reward function, in which $r_i \in R$ shows the current immediate reward for the state transition from state s into s' as a result of action a.

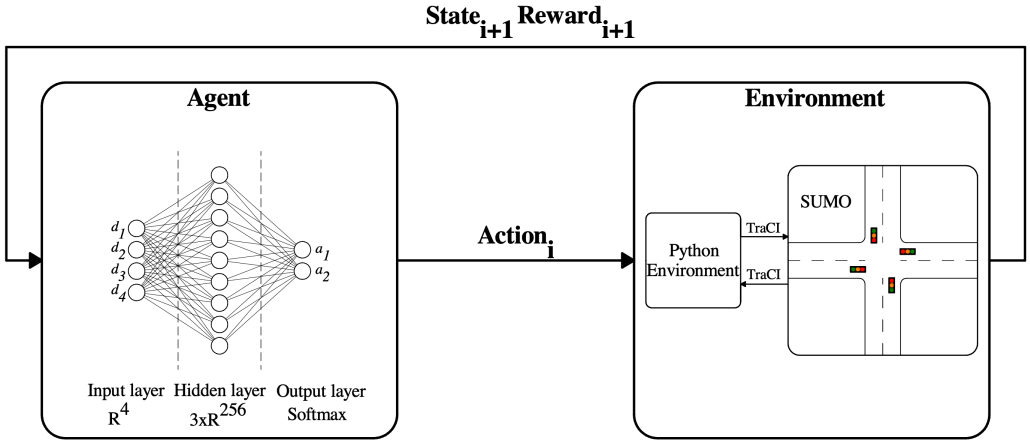

Figure 1. Reinforcement learning cycle.

RL can be formulated as an episodic setting where the agent interacts with the environment at every given discrete time step until termination using the above-explained framework. It observes the state of the environment $s \in S$ at every $t \in \{1,2,3,4,\ldots\}$, and chooses an action $a \in A$ to maximize the expected reward $r \in R$. With each step the environment transitions from state s to s'. In the long run, the agent tries to maximize the cumulative reward - based on the given reward function. Thus we can express the

goal of an RL agent as to find a policy $\pi : S \mapsto A$ that maximizes the value of every state-action pair:

$$Q^{\pi}(s,a) \equiv \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^{i} r(S_{t+i}, A_{t+i}, S_{t+i+1}) | S_t = s, A_t = a\right], \quad (1)$$

where $\mathbb{E}^{\pi}[\cdot]$ denotes expectation over the episode induced by π, r is the reward function, conditioned on the current and next state and the action causing the transition. Furthermore, $\gamma \in [0,1)$ is the discount factor used for setting how time deteriorates the importance of rewards.

2.2. Artificial Neural Network Representation

In general, deep learning and deep reinforcement learning use ANNs to approximate complex functions. We considered the most straightforward network layers to both of our proposed agents, especially Multi-Layer Perceptrons (MLP). A one-layer MLP stands from a feed-forward fully connected linear layer and some kind of non-linearity, e.g., ReLU in our case. The linear layer has bias and performs a linear transform on its input. Basically, this is a matrix multiplication and provides an output according to (2).

$$y = xA^{T} + b, \quad (2)$$

where x denotes the input, y stands for the output, A contains the weight matrix of the layer, and b is the used bias.

Since all nodes are connected with each other, and arbitrary output dimensions are achievable, usually Linear is used for the output of a regression model. The non-linearity layer is an activation function with the characteristics: $ReLu(x) = max(0, x)$. With this characteristic, only the positive values are passed through the layer, filtering the negative values and thus their undesirable and unwanted properties. [35].

- in the case of the DQN agent, the last layer is the output of the second layer of the MLP without activation.
- in the case of PG as the last layer of the network $Softmax$ is included because it is a normalizing function, providing a normalized output between $[0,1)$ with the sum of the output equal to 1.

The above considerations are taken to have a general internal structure of the agents, however, satisfying the needs of the algorithms described later in this section. As activation layer, *ReLu* was chosen while considering its positive attributes and utilizing its convergence supporting features.

2.2.1. Policy Gradient

The Vanilla Policy Gradient (PG) algorithm is a member of the basic policy-based algorithms applied in RL. Policy-based methods earned the interest of researchers with their tremendous results in a variety of domains [36]. Moreover, the PG algorithm has an essential advantage over the value-based methods, notably that it is guaranteed that it converges at least to a local optimum [37]. In policy-based methods, the neural network weights are tuned to approximate a probability distribution over the executable actions for every given state. Thus, the predicted values can be interpreted as dynamic heuristics since the probabilities do not reveal how each choice benefits the agent in the long run. In our work, to support better exploration, a categorical function is created based on these probability output values, and a sampling is carried out to defines the final action the agent will proceed with. Algorithms 1 and 2 shows the pseudo-code for the policy gradient method.

Algorithm 1 Vanilla Policy Gradient.

```
procedure VANILLAPOLICY(EPOCH)
    Initialize (θ_0), the weights of initial policy
    for episode in range of epoch do
        while not terminated do
            Sample action based on current Policy
            Step one in the environment
            Save new state, reward and actual policy
        updatePolicy()
    Return Policy
```

Algorithm 2 UpdatePolicy.

```
procedure UPDATEPOLICY()
    Calculate loss = −Σ_{i=0}^{e_length} r_i · log_2(P(a_i|π_i))
    Compute gradients
    if update then
        Step with optimizer→ (θ_t)
        Zero out gradients
    else
        Continue
```

In case of PG the loss is calculated with the following equation:

$$loss = \sum_{i=0}^{t} r_i \cdot \log_2(P(a_i|\pi_i)), \tag{3}$$

where

$$a_i \in \mathbf{A}, i \in \{0, 1, 2, \ldots, t\}$$

Here the r_i is the given reward at ith time step achieved by action a_i applied in the state input s, the $\log_2(P(a_i|\pi_i))$ means the probability of choosing action a in a given π_i and s.

2.2.2. Deep Q-Learning

The first and most substantial breakthroughs of RL are strongly tied to the value-based methods such as DQN [30], Double-DQN [38], and Dueling-DQN [39]. These were the first results that showed that trained agents could operate on the same level as humans and outperform them in some cases. In these methods, the neural network weights are tuned to approximate the amount of cumulated reward that the agent can score with the executable actions in the given scenario during the episode. To do so, the Bellman equation is used as an update rule to improve the predicted Q values:

$$Q(s_t, a_t; \theta_t) = r_{t+1} + \gamma \max_a Q(s_{t+1}, a_t; \theta_t^-). \tag{4}$$

Compared to the policy-based methods, these values can be interpreted as absolute heuristics since each value shows the benefits of a choice in the long run. The decision-making strategy of the training process also differs from that of PG since it utilizes an *e–greedy* approach. Moreover, DQN is considered an indirect method contrary to PG. In the case of PG, the probabilities directly suggest a policy that emerges into a behavior, while the predicted values of the DQN agents are more like a situation interpretation has to be exploited with the suitable policy. Algorithm 3 shows the pseudo-code for the DQN algorithm.

Algorithm 3 Deep Q-learning with replay memory.

procedure TRAIN
  Initialize parameters, weights θ, θ_{target}
  for each episode **do**
    $s_t \leftarrow$ reset environment state ▷ $t = 0$
    while not done **do**
      if Bernoulli(ϵ) = 1 **then**
        $a_t \leftarrow Uniform(\mathbf{A})$
      else
        $a_t \leftarrow \underset{b \in A}{\text{argmax}} \, (Q^{\pi}(s_t, b), \quad \forall b \in A$
      $s_{t+1}, r_t, done \leftarrow Env(a_t)$ ▷ step with a_t
      $memory \leftarrow s_t, a_t, s_{t+1}, r_t$, done
      $s_t \leftarrow s_{t+1}$
    UPDATE_WEIGHT()

2.3. Baseline Controller

Some papers utilize different RL solutions [20,21], actuated controllers [17], and fixed time controllers [18,19,22] for baseline to evaluate the performance of their solutions in various measures. This work compares different RL algorithms trained with our novel rewarding concept and utilizes actuated control for baseline. We omit fixed-time control because it can not handle unbalanced changes in the traffic volume; hence it can be outperformed easily. The utilized actuated control is SUMO's built-in Time-loss based actuated controller that considers the time loss of the vehicles in the lane as the basis of its behavior.

3. Environment

As discussed before, the RL framework requires an environment to interact. In our case, Simulation of Urban MObility (SUMO) [40] serves the purpose of a simulation environment. SUMO is an open-source, microscopic, and macroscopic traffic simulator that can be extended dynamically and highly customizable. One can build different networks or even load entire networks through the APIs developed by the community. As Figure 1 displays, our infrastructure looks as follows: The PG and DQN agents are implemented in python with the use of the Pytorch Deep Learning library. In the meantime, the environment consists of two major parts, a python component and the intersection created in SUMO , see Figure 2. The python component is responsible for manipulating the SUMO environment based on the agent's action with the TraCI interface's help and gathering the information for state representation, rewarding, and evaluating the performance. Moreover, the simulations can be easily randomized, making it an excellent choice for RL since diverse experiences are required for a robust outcome in the training process.

In our setting, there are four road segments, which meet at one intersection (Figure 2). All segments have 500 m lengths and contain two-direction lanes, each 3.2 m wide. At the four ends, we added vehicle departure points, where new vehicles enter the simulation at each step randomly. In every episode we select a frequency of launching a new vehicle at each end uniformly in the interval of $[1,\ 200] \frac{dt}{vehicle}$. This means that in case of frequency 1, at every simulation time step (dt) a vehicle will be launched at that entry point.

For evaluation purposes, we use some of the SUMO simulator features that help us measure the impact of traffic on the environment. With the tool, we extract measures such CO_2, CO, NO_x, HC, PM_x, and fuel consumption. As a result, the different solutions for the Traffic Signal Control (TSC) problem can be compared on an environmentally friendly scale, beyond the classic flow measures, as detailed in Section 4.

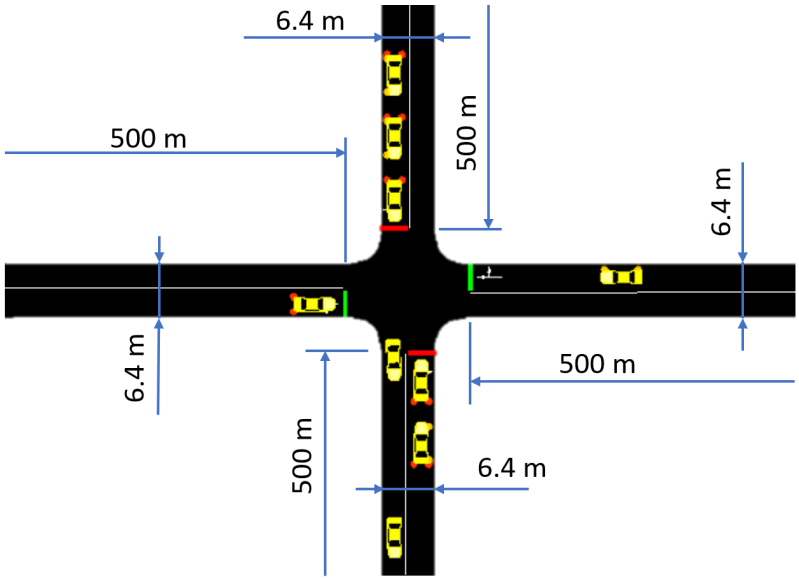

Figure 2. Layout of the intersection.

3.1. Limitations

The utilized network type in this paper is a single isolated intersection, where the potential role of pedestrians and pedestrian crossings along with bicyclists is not considered. Furthermore, in the simulation, there are only passenger cars. Hence the formulated environment does not prioritize road users, such as transit vehicles, emergency vehicles, or buses. Consequently, these aspects are not considered during the formulation of the solution. The exclusive usage of passenger cars in the simulation is also important from the emission standpoint since SUMO's emission model differentiates based on vehicle types. In the intersection, the left turns are omitted to avoid the additional congestion generated by the time spent deciding which vehicle should go first. Thanks to that, the agent can clearly observe the impact of its actions on the intersection, which further accelerates the learning process. Its noise-reductive effect, no delays in traffic due to negotiations, also supports the precise evaluation of the utilized abstraction for training. The goal of the launched vehicles is also based on a uniform distribution, resulting in an equal number of cars turning right and going straight at the intersection. The intersection is controlled through a traffic light, whose optimal manipulation will be the agents' task.

3.2. State Representation

In the RL framework, one has two duties regarding the environment. These are the formulation of the state representation and the rewarding concept. These two components are responsible for the proper credit assignment. The formulation of the state representation is basically the abstraction of the control problem, which provides the influential features of the task. Hence, the researcher's intuitions decide whether the agent can utilize the potential of RL or not. In the TSC problem, two main approaches are used for state representation. The first is raw pixel data in the form of snapshots from the intersection, which require relatively more computation resources since the use of convolutional neural networks (CNN) [41,42]. The second is feature-based value vectors which contain information about each lane provided by simple detectors [43,44]. This information can be the average speed of the vehicles, queue length, waiting time, and so forth. The presented research utilizes the second approach and represents each lane is with a single value which is the occupancy of the given 500 m long road. These occupancy values are transformed into the [0, 1] interval for avoiding numerical problems of the back-propagation method.

3.3. Action Space

There are two different types of actions for the TSC problem. In the first case, one can change the duration of the phases in a predefined interval continuously. In the second case, used by this paper, changes the order of the traffic light phases after the previous phase duration is ended [45]. There are also hybrid approaches, which combine the two former concepts [22]. This paper uses the second approach. There are two main phases:

- The first one, called East-West Green (EWG), provides a green signal for the pair of vertical lanes and red for the horizontal ones.
- The second phase, called North-South Green (NSG), allows the horizontal lanes to pass and signals red to the vertical ones.

With the settings of these two phases, the agent can prolong the green light for the lanes. The duration of each phase is at least 30 s. It is worth mentioning that every phase change is separated with a yellow, yellow-green phase, respectively. This approach ensures the agent understanding the consequences of its actions and lets the vehicles adjust to the new state in the environment. It is also important to note that the simulations are generated randomly with defined random vehicle flows at every episode, which gives the power to the agent to generalize on different situations. As the different flows result in diverse loads on the road segments, the agent must carefully pay attention to the lamp states and find the right changing points to have the desired throughput in the intersection.

3.4. Rewarding Concept

The rewarding concept is one of the most crucial aspects of RL since the agent only uses these feedback values to understand the inner dynamics of the process, which is the core of credit assignment. Thanks to that, the rewarding concept can firmly influence the success of the training and thus the agent's performance. In the literature, most of the authors utilize waiting time and delay-based rewarding concepts [46] or queue length, discharge [47]. Compared to these rewarding schemes, a novel strategy is presented.

Our approach relies on the occupancy of the oncoming lanes, which are interpreted as a distribution. The introduced actions (NSG, EWG) can control the mean and standard deviation of the occupancy-based distribution by providing more green time to one or another incoming lane. Considering this aspect of the TSC problem, the reward of each step is calculated from the occupancy distribution as its standard deviation, normalized to the $[-1;1]$ interval. The standard deviation is chosen because it aims to balance the provided green time between the incoming lanes of the intersection according to the traffic flow that occurred on them. The proposed rewarding looks formally as follows:

$$R = R_{max} + \frac{(\sigma - \sigma_{min} * (R_{min} - R_{max}))}{(\sigma_{max} - \sigma_{min})} \tag{5}$$

The parameters in (5) depends on the standard deviation σ, which is calculated from the occupancy of the lanes. The value choices are shown in Table 1.

Table 1. Parameter choices for different σ values.

Parameter	$\sigma < 0.1$	$\sigma \geq 0.1$
R_{max}	1	0
R_{min}	0	-1
σ_{max}	0.1	0.5
σ_{min}	0	0.1

Generally, in the formulation of the reward concept, a great deal of thought must be focused on the learnable loopholes, because such pitfalls of the rewarding might ruin the final performance of the trained model. In our case, as a possible loophole scenario can happen where only a few vehicles stay in the horizontal or vertical lanes, and the agent

provides green time to the direction with absolutely no traffic flow. Since in the above described scenario the standard deviation is small, the agent would get a positive reward from the environment despite the fact that provides green time to the wrong lanes. This makes our control problem impossible to solve. This loophole is neutralized by punishing the agent with a -1 scalar feedback value, when it provides green time to a lane without traffic flow. Nevertheless, the episode is not terminated in such cases because without termination much more experience can be gathered about this particular scenarios and that promotes the learning and generalization.

3.5. Important Aspects of Real-World Implementation

When formulating an RL solution for the TSC problem, the possibility of real-world implementation should also be considered. In the case of RL-based algorithms, it means choosing easily accessible traffic measures for the state representation and the reward strategy. In modern cities, there are a lot of sensors that are used for collecting information about the traffic flow, congestions, accidents, occupancy of lanes, and queue lengths in intersections. These sensors are loop detectors, magnetic sensors, cameras, lidar sensors, probe vehicle sensors, mobile phones, wi-fi, Bluetooth, and so forth. The management of the collected large amount of data is also can be done in several ways. Still, a promising contender is V2X communication combined with 5G [48], thanks to its short latency. This paper tries to create a representation and a rewarding system that supports real-world implementation through easily accessible features. Our concept only requires the occupancy of the incoming lanes. The occupancy of lanes can be monitored with simple loop-detectors, which are an affordable and reliable choice, not to mention that it is already installed in larger cities worldwide.

3.6. Training

The training scenario has vital importance in the success of the agent. It has to provide a diverse set of experiences to the agent to reach a sufficient level of robustness, hence the best possible performance and generalized solution. The training is realized with the introduced SUMO environment, with which it is possible to randomize the traffic flow for every episode. This means different drivers are launching from the departure mentioned above points with random speed profiles and route destinations. Thus, in each episode, a predefined number of vehicles enter the simulation. Thanks to this randomization, the agents have the opportunity to solve a great variety of traffic situations with diverse use-cases. Every training episode starts with a warm-up stage, where vehicles start to enter the network, and it lasts until the summarized occupancy of all four lanes reaches 10%. During the warm-up time interval, all traffic lights are set to red. If the warm-up stage has ended, the agent controls the traffic lights with the introduced actions. A training episode lasts until all the vehicles pass through the intersection. If the agent fails to pass through all the vehicles, the episode is stopped after 2000 steps (s) measured in the SUMO simulation time. The convergence of the two RL solutions is shown in Figure 3. It shows the average reward gathered throughout the episodes. Based on the convergences' characteristics, it naturally comes that the PG algorithm is more stable and has better convergence properties such as speed over the DQN algorithm. This might be because DQN methods can overestimate the Q values of some actions, and thus when selecting the best action, the deviation of the resulting rewards can differ significantly from their beliefs. However, as it can be read from the figure, the training converges to the optimal. Based on this figure, we expect that the PG algorithm outperforms the DQN in the evaluation stage in all measures.

One of the most critical hyperparameters is the learning rate because it profoundly influences the convergence of the model. A high learning rate resulted in a suboptimal solution; furthermore, a small one slowed down the convergence for both agents. In general, the PG agent requires a lower learning rate because the algorithm tends to become

deterministic in the case of huge gradient steps. In some cases, the learning rate caused the agent to be stuck at a local minimum, hence never finding the global optimum.

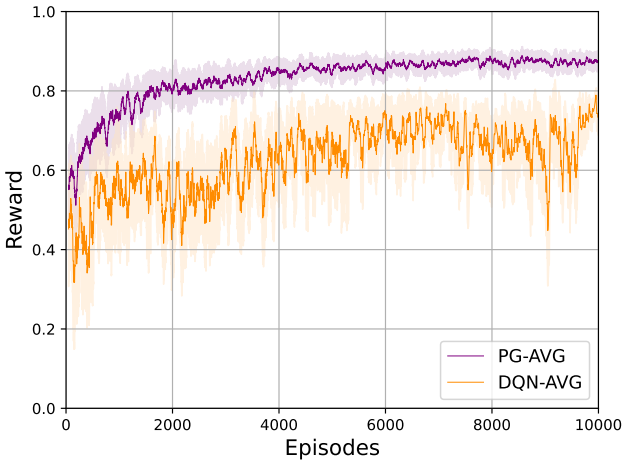

Figure 3. Convergence of the PG and DQN algorithms.

4. Results

Today, keeping our planet healthy is a significant concern and a leading goal. Consequently, a lot of attention is directed to minimizing the environmental footprint of transportation. When testing new traffic control algorithms, it comes naturally to investigate their performance against sustainability measures, along with traditional measures such as travel time, waiting time, and queue length.

For the sake of comparability, all the solutions are evaluated on 1000 episodes, where the traffic flow of the incoming lanes is randomized. Still, for each agent, the same settings are applied. Contrary to training episodes, where the episodes terminate when all vehicles pass through the intersection. The test episodes last until all the vehicles have left the network.

4.1. Comparison of the Sustainability Measures

Tables 2 and 3 show the performance of different solutions for the TSC problem on a diverse set of emission measures, e.g., carbon-dioxid, carbon-oxid, nitrogen-oxid and the fuel consumption. The different values are calculated by SUMO, and the mean of the values over the 1000 episodes are shown in the Tables 2 and 3. SUMO utilizes The Handbook Emission Factors for Road Transport (HBEFA) model for emission calculation. This model calculates the vehicles' emissions based on polynomes derived from the velocity and acceleration of the vehicle. This approach distinguishes between vehicle type and emission categories also, for more information [49].

Table 2. Statistical comparison of the sustainability measures in 1000 consecutive runs.

Agent	CO_2 Emission (kg)	CO Emission (kg)	NO_x Emission (g)
PG	66.7	2.2	29.1
DQN	70.9	2.3	31.1
Loss-Based	73.8	2.6	32.3

Table 3. Statistical comparison of the sustainability measures in 1000 consecutive runs.

Agent	PM_x Emission (g)	HC Emission (g)	Fuel Consumption (L)
PG	1.4	11.4	28.7
DQN	1.5	12.6	30.5
Loss-Based	1.6	13.7	31.7

The lower the value, the better the method performs. It can be seen that the PG agent outperforms the others in all of the emission-related measures. The second most sustainably controlling solution is our DQN. As mentioned earlier, one of the essential emission categories from the aspect of global warming is CO_2. Thanks to the proposed rewarding concept, the PG agent can save 9% percentage of the CO_2 emission compared to the actuated control. Another critical measure is the fuel consumption along with the test episodes. Our solution decreased the average fuel consumption by approximately 9%, which also has a profound environmental impact in the long run.

To further investigate the performance of the different solutions in the sustainability metrics, the CO_2 emission and fuel consumption are compared on [10, 50, 100, 500, 1000, 5000, 10,000] random episodes for every run. In this way, we can test the consistency of the algorithm's ability. Figure 4 shows the CO_2 emission results, and Figure 5 depicts the fuel consumption throughout the different number of episodes. Both Figures show the max, min, and average values along with the distribution of the individual episode averages for all three algorithms and every run. In all the runs, the RL-based solutions have lower max and average emission and fuel consumption values than the time loss-based actuated control except the DQN's performance in the first run, where the time loss-based approach has better performance. However, still, the PG agent has superiority in this and every other run.

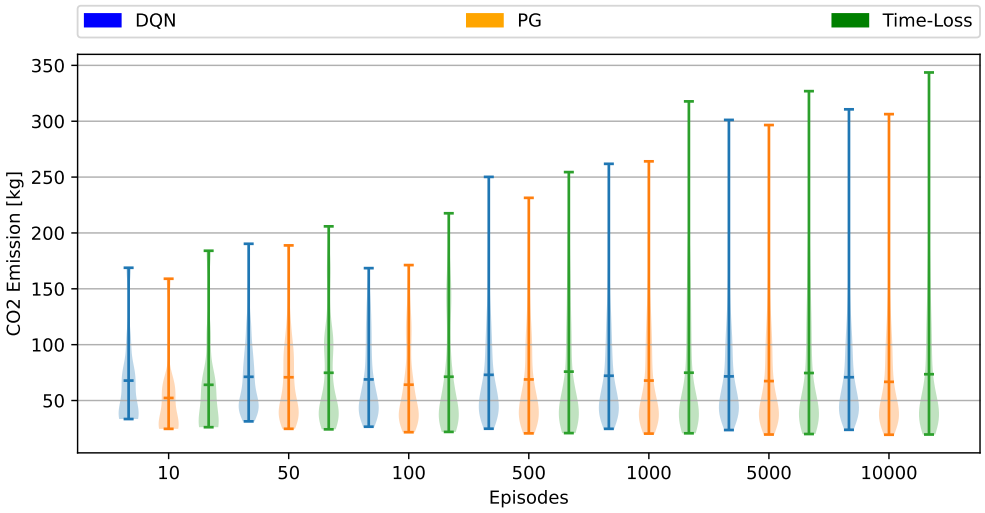

Figure 4. Distribution of CO_2 emission in different amount of random episodes.

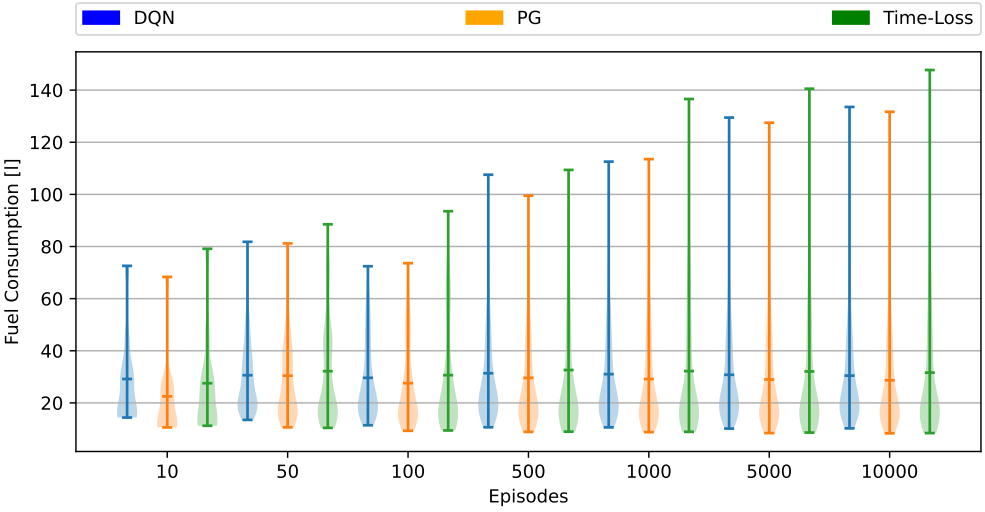

Figure 5. Distribution of Fuel consumption in different amount of random episodes.

Considering that the applied rewarding concept does not utilize any sustainability-related metrics, the agent's performance on the measures can be a consequence of the smaller conjunction and better traffic flow in the intersection. To support our hypothesis Table 4 shows how the average travel time, average waiting time, and queue length have developed over the mentioned 1000 episodes.

Table 4. Statistical comparison of the performance measures in 1000 consecutive runs.

Agent	Travel Time (s)	Waiting Time (s)	Queue Length (No. of Veh.)
PG	141.3 ± 42	29.4 ± 21	10.8 ± 8
DQN	150.7 ± 37	37.8 ± 22	13.2 ± 8
Loss-Based	141.7 ± 45	46.8 ± 49	16.1 ± 16

4.2. Comparison of the Classic Measures

Travel time is defined as the time that a vehicle is needed to arrive at its destination from the moment of departure. Average travel time is the sum of all vehicle travel time entered into the network divided by the number of vehicles. Formally it looks as follows:

$$ATT = \frac{1}{N_{veh}} \sum_{j=0}^{N_{veh}} t_{j,start} - t_{j,end}, \tag{6}$$

where ATT is the average travel time, and N_{veh} is the number of vehicles. $t_{j,start}$ marks the departure time of the jth vehicle and $t_{j,end}$ shows the arrival time of the jth vehicle.

The waiting time is the summarized standing time from its departure until passing through the intersection. A vehicle is considered to be in a waiting state if its speed is less than 0.1 (m/s). This measure is calculated for all vehicles that entered the network. The average waiting time is calculated as the sum of the accumulated waiting time of all vehicles, divided by the number of all vehicles. It yields to:

$$AWT = \frac{1}{N_{veh}} \sum_{j=0}^{N_{veh}} WT_j, \tag{7}$$

where AWT is the average waiting time and WT_j is the accumulated waiting time of the jth vehicle in the moment of leaving the intersection.

Queue length is calculated as the number of vehicles waiting in the given lane. For simplicity, the queue length of all lanes is summarized into one value. This results in one metric, representing the queuing in the network. Equation (8) shows:

$$QL = \sum_{l=0}^{L} q_l, \tag{8}$$

where QL is the summarized queue length of all lanes, and q_l is the queue length in lane l.

The results in Table 4 demonstrate that the PG agent performance has absolute superiority in waiting time and queue length over the actuated control and the DQN agent as well. In travel time, the actuated control outperforms the DQN agent and reaches nearly the same performance as the PG agent, but still, the PG algorithm slightly performs better. The difference between the actuated control and the PG algorithm is firm if the standard deviation of the results is considered since it shows that the PG agent conducts the traffic signal control in a more balanced manner. By monitoring the operation of the different solutions, we noticed that the main difference between the algorithms is that the actuated control utilizes longer green phases, which trigger the discrepancy in waiting time and queue length and also in the standard deviation of the travel time.

Figure 6 shows the characteristics of some key metrics (CO_2 emission, Queue length, Travel time, Waiting time) during the 1000 episode evaluation phase. These figures support

the previous ascertainment about the behavior and performance of the different solutions. Note that the presented RL solutions outperform the actuated control in almost all the episodes except the DQN agent in the case of travel time. Nevertheless, the DQN agent conducts its control in an environmentally sustainable manner along with the PG agent.

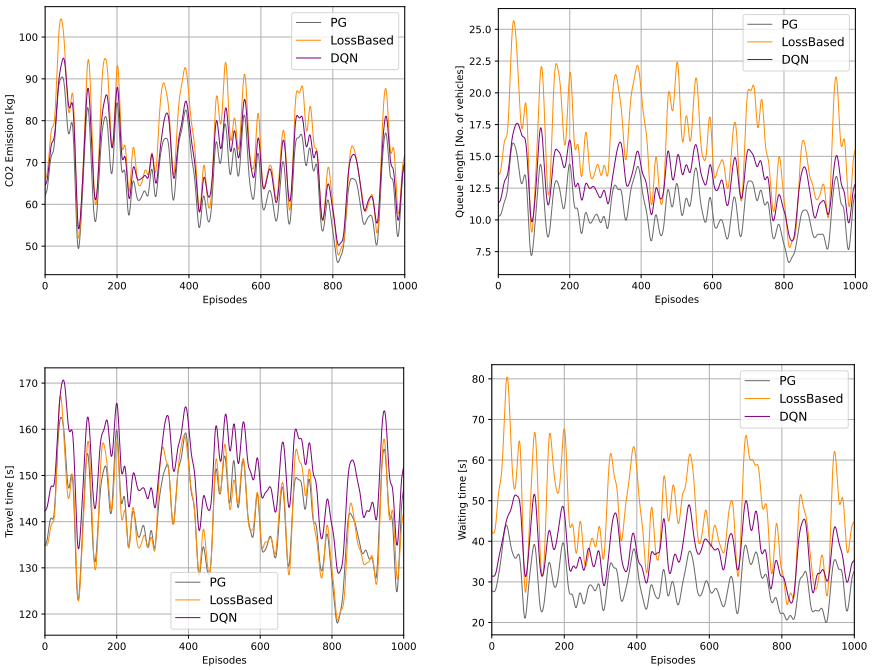

Figure 6. Performance of the agents in the different sustainability measures along the 1000 random episodes.

Waiting time strongly impacts the satisfaction of the individual traffic participants, so the proposed solutions' consistency is further evaluated. For that purpose, the total waiting time is monitored for [10, 50, 100, 500, 1000, 5000, 10,000] episodes. In each run, a different random seed is applied for all three solutions. In an episode, approximately 150–300 vehicles commute to the intersection, and the total time of all vehicles completing their journey is around 10–20 min, depending on the controlling algorithm and the number of vehicles deployed. The results of the accumulated waiting times are shown in Figure 7. The figure states that both RL solutions with the proposed rewarding concept reach a lower amount of total waiting time, hence outperform the Time-Loss based actuated controller. The difference between the RL solutions is also substantial, favoring the PG algorithm, which can maintain its supremacy. Thanks to our approach, higher satisfaction can be reached among the road users, which is also advantageous since it can decrease the stress of the individuals.

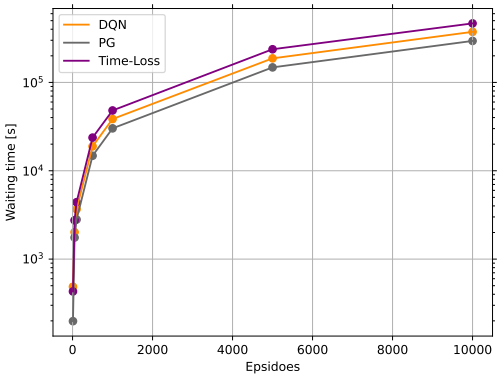

Figure 7. Comparison of total waiting time.

These results show that the RL agents trained with the new standard deviation-based reward concept reach better performance in sustainability and classic measures without any reward shaping. This suggests that we have found an interesting insight into the TSC problem's inner dynamics, which results in superior performance.

5. Conclusions

In this paper, a new rewarding concept is introduced for the single-intersection traffic signal control problem. For simplicity, the left turns are omitted, and the turning ratio is uniformly distributed between the exits of the intersection. Furthermore, the actions chosen by the agent can not cause conflicts in the intersection. Consequently, the agent's goal is to find the phase combination that results in the environmentally most sustainable control and mitigates individual drivers' waiting time and travel time along with the queue length in the intersection. A DQN and a PG agent are trained for solving the control task at hand. Both agents' behavior is assessed according to classic performance measures concerning average travel time, average waiting time, and queue length. Moreover, the trained agents are also evaluated according to sustainability measures, such as CO_2 emission, fuel consumption, NO_x emission, and so forth. The results suggest that both agents can outperform the SUMO's built-in time loss-based actuated control in every sustainability measure with the introduced rewarding concept. In the classic measures, average waiting time and queue length, both the PG and the DQN agent can go beyond the performance of the actuated control, and only the PG algorithm can perform slightly better than the actuated one in average travel time. The main advantage of our approach is that it naturally tries to keep the balance between the incoming lanes. Beyond the statistical measures, it reaches higher satisfaction of the road users since the waiting time is generally less for one individual. Another important aspect is that this approach can handle intersections, where the importance of the incoming lanes is different. This differentiation can be easily achieved by scaling the occupancy of the lanes with constants. However, the importance of a lane does not necessarily have to be predefined. It can be calculated dynamically by scaling the occupancy of the lanes according to the waiting time occurring, queue length, or any favored measure, for that matter. Thus, our solution can have plenty benefits in real-world applications. Using our algorithm for traffic signal control can indeed reduce the emission of the urban areas, as well as it can spare us time spent in traffic jams. Our agents naturally support the minimization of the congestion by the throughput maximization of the intersection. If we apply prioritization to any incoming lanes (e.g., main roads vs complimentary roads) the traffic flow can be adjusted even further. Moreover, the needed sensory requirements for the proposed state representation of our agents can be produced with loop-detectors that are cheap and reliable and also are implemented in most of the big cities around the world. Thus, we propose the experiments with our method in some real intersections that would provide useful statistics of our theory in practice. Based on the results acquired in our research about the potential of the novel standard deviation-based rewarding concept for the TSC problem, it seems reasonable to continue this research and scale up the control problem's complexity by using interdependent multi-intersections. We plan to use our novel rewarding combined with Multi-Agent Reinforcement Learning (MARL) in our future endeavors. The MARL framework possesses essential features such as communication and even negotiation between agents that undoubtedly help to solve complex, real-world problems.

Author Contributions: Conceptualization, T.B. and S.A.; methodology, P.G. and B.K.; software, B.K. and L.S.; validation, T.B and S.A.; investigation, T.B.; resources, P.G.; writing—original draft preparation, B.K. and L.S.; writing—review and editing, T.B., S.A. and P.G; visualization, B.K.; supervision, P.G.; funding acquisition, P.G. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by the Ministry of Innovation and Technology NRDI Office within the framework of the Autonomous Systems National Laboratory Program. The research was also supported by the Hungarian Government and co-financed by the European Social Fund through

the project "Talent management in autonomous vehicle control technologies" (EFOP-3.6.3-VEKOP-16-2017-00001).

Institutional Review Board Statement: Not applicable

Informed Consent Statement: Not applicable

Data Availability Statement: The data and source is available at the authors.

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

Abbreviations

The following abbreviations are used in this manuscript:

DRL	Deep Reinforcement Learning
TSC	Traffic Signal Control
IEA	International Energy Agency
ITS	Intelligent Transportation Systems
GA	Genetic Algorithm
DP	Dynamic Programing
MAS	Multi-Agent Systems
DQN	Deep Q-Network
PG	Policy Gradient
CNN	Convolutional Neural Networks
DL	Deep Learning
ANN	Artifical Neural Networks
SL	Supervised Learning
MDP	Markov Decision Process
MLP	Multi-Layer Perceptron
SUMO	Simulation of Urban MObility
EWG	East West Green
NSG	North South Green
HBEFA	Handbook Emission Factors for Road Transportation
ATT	Average Travel Time
AWT	Average Waiting Time
QL	Queue Length

References

1. Al-Ghussain, L. Global warming: Review on driving forces and mitigation. *Environ. Prog. Sustain. Energy* **2019**, *38*, 13–21. [CrossRef]
2. Acar, C.; Dincer, I. The potential role of hydrogen as a sustainable transportation fuel to combat global warming. *Int. J. Hydrogen Energy* **2020**, *45*, 3396–3406. [CrossRef]
3. Hannappel, R. The impact of global warming on the automotive industry. *AIP Conf. Proc.* **2017**, *1871*, 060001.
4. Ekeila, W.; Sayed, T.; Esawey, M.E. Development of dynamic transit signal priority strategy. *Transp. Res. Rec.* **2009**, *2111*, 1–9. [CrossRef]
5. Dell, P.; Mirchandani, B. REALBAND: An approach for real-time coordination of traffic flows on networks. *Transp. Res. Rec.* **1995**, *1494*, 106–116.
6. Wang, F.; Tang, K.; Li, K.; Liu, Z.; Zhu, L. A group-based signal timing optimization model considering safety for signalized intersections with mixed traffic flows. *J. Adv. Transp.* **2019**, *2019*, 2747569. [CrossRef] [PubMed]
7. Stevanovic, J.; Stevanovic, A.; Martin, P.T.; Bauer, T. Stochastic optimization of traffic control and transit priority settings in VISSIM. *Transp. Res. Part C Emerg. Technol.* **2008**, *16*, 332–349. [CrossRef]
8. Zhang, L.; Yin, Y.; Chen, S. Robust signal timing optimization with environmental concerns. *Transp. Res. Part C Emerg. Technol.* **2013**, *29*, 55–71. [CrossRef]
9. McKenney, D.; White, T. Distributed and adaptive traffic signal control within a realistic traffic simulation. *Eng. Appl. Artif. Intell.* **2013**, *26*, 574–583. [CrossRef]
10. Boillot, F.; Midenet, S.; Pierrelee, J.C. The real-time urban traffic control system CRONOS: Algorithm and experiments. *Transp. Res. Part C Emerg. Technol.* **2006**, *14*, 18–38. [CrossRef]
11. Cai, C.; Wong, C.K.; Heydecker, B.G. Adaptive traffic signal control using approximate dynamic programming. *Transp. Res. Part C Emerg. Technol.* **2009**, *17*, 456–474. [CrossRef]

12. Mirchandani, P.; Head, L. A real-time traffic signal control system: Architecture, algorithms, and analysis. *Transp. Res. Part C Emerg. Technol.* **2001**, *9*, 415–432. [CrossRef]
13. Jin, J.; Ma, X. A group-based traffic signal control with adaptive learning ability. *Eng. Appl. Artif. Intell.* **2017**, *65*, 282–293. [CrossRef]
14. Villalobos, I.A.; Poznyak, A.S.; Tamayo, A.M. Urban traffic control problem: A game theory approach. *IFAC Proc. Vol.* **2008**, *41*, 7154–7159. [CrossRef]
15. Fehér, Á.; Aradi, S.; Bécsi, T. Fast Prototype Framework for Deep Reinforcement Learning-based Trajectory Planner. *Period. Polytech. Transp. Eng.* **2020**, *48*, 307–312. [CrossRef]
16. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Al Sallab, A.A.; Yogamani, S.; Pérez, P. Deep reinforcement learning for autonomous driving: A survey. *IEEE Trans. Intell. Transp. Syst.* **2021**. [CrossRef]
17. Wang, S.; Xie, X.; Huang, K.; Zeng, J.; Cai, Z. Deep reinforcement learning-based traffic signal control using high-resolution event-based data. *Entropy* **2019**, *21*, 744. [CrossRef]
18. Guo, J.; Harmati, I. Comparison of Game Theoretical Strategy and Reinforcement Learning in Traffic Light Control. *Period. Polytech. Transp. Eng.* **2020**, *48*, 313–319. [CrossRef]
19. Wan, C.H.; Hwang, M.C. Value-based deep reinforcement learning for adaptive isolated intersection signal control. *IET Intell. Transp. Syst.* **2018**, *12*, 1005–1010. [CrossRef]
20. Li, L.; Lv, Y.; Wang, F.Y. Traffic signal timing via deep reinforcement learning. *IEEE/CAA J. Autom. Sin.* **2016**, *3*, 247–254.
21. Genders, W.; Razavi, S. Using a deep reinforcement learning agent for traffic signal control. *arXiv* **2016**, arXiv:1611.01142.
22. Bouktif, S.; Cheniki, A.; Ouni, A. Traffic signal control using hybrid action space deep reinforcement learning. *Sensors* **2021**, *21*, 2302. [CrossRef]
23. Tan, T.; Bao, F.; Deng, Y.; Jin, A.; Dai, Q.; Wang, J. Cooperative deep reinforcement learning for large-scale traffic grid signal control. *IEEE Trans. Cybern.* **2019**, *50*, 2687–2700. [CrossRef]
24. Van der Pol, E.; Oliehoek, F.A. Coordinated deep reinforcement learners for traffic light control. In Proceedings of the Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016), Barcelona, Spain, 5 December 2016.
25. Chu, T.; Wang, J.; Codecà, L.; Li, Z. Multi-agent deep reinforcement learning for large-scale traffic signal control. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 1086–1095. [CrossRef]
26. Eom, M.; Kim, B.I. The traffic signal control problem for intersections: A review. *Eur. Transp. Res. Rev.* **2020**, *12*, 1–20. [CrossRef]
27. Farazi, N.P.; Ahamed, T.; Barua, L.; Zou, B. Deep Reinforcement Learning and Transportation Research: A Comprehensive Review. *arXiv* **2020**, arXiv:2010.06187.
28. Wei, H.; Zheng, G.; Gayah, V.; Li, Z. Recent advances in reinforcement learning for traffic signal control: A survey of models and evaluation. *ACM SIGKDD Explor. Newsl.* **2021**, *22*, 12–18. [CrossRef]
29. Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. Mastering the game of go without human knowledge. *Nature* **2017**, *550*, 354–359. [CrossRef] [PubMed]
30. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
31. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
32. Aradi, S. Survey of deep reinforcement learning for motion planning of autonomous vehicles. *IEEE Trans. Intell. Transp. Syst.* **2020**. [CrossRef]
33. Schrittwieser, J.; Antonoglou, I.; Hubert, T.; Simonyan, K.; Sifre, L.; Schmitt, S.; Guez, A.; Lockhart, E.; Hassabis, D.; Graepel, T.; et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature* **2020**, *588*, 604–609. [CrossRef] [PubMed]
34. Sutton, R.S.; Barto, A.G. *Introduction to Reinforcement Learning*, 1st ed.; MIT Press: Cambridge, MA, USA, 1998.
35. Nair, V; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning (ICML 2010), Haifa, Israel, 21–24 June 2010.
36. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1928–1937.
37. Sutton, R.S.; McAllester, D.A.; Singh, S.P.; Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2000; pp. 1057–1063.
38. Van Hasselt, H.; Guez, A.; Silver, D. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
39. Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; Freitas, N. Dueling network architectures for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1995–2003.
40. Lopez, P.A.; Behrisch, M.; Bieker-Walz, L.; Erdmann, J.; Flötteröd, Y.; Hilbrich, R.; Lücken, L.; Rummel, J.; Wagner, P.; Wiessner, E. Microscopic Traffic Simulation using SUMO. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2575–2582. [CrossRef]
41. Garg, D.; Chli, M.; Vogiatzis, G. Deep reinforcement learning for autonomous traffic light control. In Proceedings of the 2018 3rd IEEE International Conference on Intelligent Transportation Engineering (ICITE), Singapore, 3–5 September 2018; pp. 214–218.

42. Mousavi, S.S.; Schukat, M.; Howley, E. Traffic light control using deep policy-gradient and value-function-based reinforcement learning. *IET Intell. Transp. Syst.* **2017**, *11*, 417–423. [CrossRef]
43. Liang, X.; Du, X.; Wang, G.; Han, Z. A deep reinforcement learning network for traffic light cycle control. *IEEE Trans. Veh. Technol.* **2019**, *68*, 1243–1253. [CrossRef]
44. Ha-li, P.; Ke, D. An intersection signal control method based on deep reinforcement learning. In Proceedings of the 2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA), Changsha, China, 9–10 October 2017; pp. 344–348.
45. Guo, M.; Wang, P.; Chan, C.Y.; Askary, S. A reinforcement learning approach for intelligent traffic signal control at urban intersections. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 4242–4247.
46. Gao, J.; Shen, Y.; Liu, J.; Ito, M.; Shiratori, N. Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network. *arXiv* **2017**, arXiv:1705.02755.
47. Muresan, M.; Fu, L.; Pan, G. Adaptive traffic signal control with deep reinforcement learning an exploratory investigation. *arXiv* **2019**, arXiv:1901.00960.
48. Szalay, Z.; Ficzere, D.; Tihanyi, V.; Magyar, F.; Soós, G.; Varga, P. 5G-enabled autonomous driving demonstration with a V2X scenario-in-the-loop approach. *Sensors* **2020**, *20*, 7344. [CrossRef]
49. Keller, M.; Hausberger, S.; Matzer, C.; Wüthrich, P.; Notter, B. *Handbook Emission Factors for Road Transport (HBEFA) Version 3.3*; Background Documentation; INFRAS: Bern, Switzerland, 2017; Volume 12.