IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 14, 2020, accepted September 4, 2020, date of publication September 18, 2020, date of current version September 30, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.3024979*

# Optimizing Transportation Dynamics at a City-Scale Using a Reinforcement Learning Framework

**LUCKYSON KHAIDEM[1], MASSIMILIANO LUCA[2,3], FAN YANG[1], (Member, IEEE), ANKIT ANAND[1], BRUNO LEPRI[2], (Member, IEEE), AND WEN DONG[1], (Member, IEEE)**

[1]Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo, NY 14260, USA
[2]Fondazione Bruno Kessler, 38123 Trento, Italy
[3]Faculty of Computer Science, Free University of Bozen-Bolzano, 39100 Bolzano, Italy

Corresponding author: Massimiliano Luca (mluca@fbk.eu)

**ABSTRACT** Urban planners, authorities, and numerous additional players have to deal with challenges related to the rapid urbanization process and its effect on human mobility and transport dynamics. Hence, optimize transportation systems represents a unique occasion for municipalities. Indeed, the quality of transport is linked to economic growth, and by decreasing traffic congestion, the life quality of the inhabitants is drastically enhanced. Most state-of-the-art solutions optimize traffic in specific and small zones of cities (e.g., single intersections) and cannot be used to gather insights for an entire city. Moreover, evaluating such optimized policies in a realistic way that is convincing for policy-makers can be extremely expensive. In our work, we propose a reinforcement learning frameworks to overtake these two limitations. In particular, we use human mobility data to optimize the transport dynamics of three real-world cities (i.e., Berlin, Santiago de Chile, Dakar) and a synthesized one (i.e., SynthTown). To this end, we transform the transportation dynamics' simulator MATSim into a realistic reinforcement learning environment able to optimize and evaluate transportation policies using agents that perform realistic daily activities and trips. In this way, we can assess transportation policies in a manner that is convincing for policy-makers. Finally, we develop a model-based reinforcement learning algorithm that approximates MATSim dynamics with a Partially Observable Discrete Event Decision Process (PODEDP) and, with respect to other state-of-art policy optimization techniques, can scale on big transportation data and find optimal policies also on a city-scale.

**INDEX TERMS** Transportation dynamics, human mobility data, reinforcement learning, partially observable discrete event decision process, MATSim.

## I. INTRODUCTION

Organizing and managing cities that are fastly becoming bigger will be one of the most critical challenges of the next decade. The urbanization process is a global challenge that involves all the continents, and it poses questions on how to make the future of mega-cities more livable and sustainable from many perspectives such as ecology, water, and energy management [1]. Also, transportation systems play an essential role in attaining this purpose, and thus redesigning and modernizing urban mobility remains a pivotal factor for our metropolitan landscapes.

The associate editor coordinating the review of this manuscript and approving it for publication was Nikhil Padhi.

Moreover, several studies have provided evidence that the quality of the transportation systems is related to the economic well-being of both small [2] and large areas [3]. Factors such as traffic jams have a consequence on the quality of life, the vitality, and the health of citizens [4]–[6]. Hence, transportation engineers are doing their best to overtake the before-mentioned concerns by using innovative solutions identified as Intelligent Transportation Systems (ITS). The primary purpose of ITS is to enhance decision-making processes for transportation-related tasks, and data can provide a considerable supplement. Nowadays, data are more accessible than ever before, thanks to the rise of ubiquitous computing, including the usage of sensors by citizens and in our cities. For instance, GPS devices installed on cars and

mobile phones are extensively used to gain an up-to-date overview of the mobility patterns in a city with a remarkably high spatial granularity [7]–[9].

The availability of such precious knowledge provides insights of paramount importance for the management of transportation dynamics and the related decision-making processes. For example, it is possible to determine the traffic state in a city [10] while other techniques allow us to forecast the volume of traffic within a specific period [11]. Another example is represented by the possibility to forecast the speed travel on a specific link [12], [13]. Notwithstanding the goodness of many state-of-the-art approaches, ITS still poses several challenges for researchers. For example, the most recent algorithms and models are often focused on solving transportation network dynamics only on specific parts of a city (see Section II).

Significant progress in transportation research has been due to simulators' role, namely, environments in which engineers can experiment with their models and gain enlightening results. Two widely used examples of simulators are (i) SUMO (Simulation of Urban MObility) [14] and (ii) MATSim [15]. The former is an open-source traffic simulator that provides APIs and a graphical interface to model road networks. It was used, for example, for defining an environment in which a deep reinforcement-learning adaptive traffic signal control agent can operate [16] and, in [17] in which the traffic optimization is extended to a multi-agent problem where each agent controls a traffic light. Instead, MATSim, recently proposed by Horni *et al.* [15], is an open-source framework for simulating transportation dynamics in a multi-agent environment that offers a wide variety of modules for demand-modeling, traffic flow simulation, and re-planning. Interestingly, this simulation framework can be extended to cover specific needs by integrating models and source codes.

In this work, we adopt the state-of-the-art high-fidelity multi-agent transportation simulator MATSim. We transform it into a reinforcement learning environment to have a scenario in which agents act realistically. In this way, we define a framework in which we can test optimized transportation policies in convincing ways for policy-makers serving as an alternative to real-world tests that are extremely expensive and, in some cases, dangerous. Moreover, thanks to agents' realistic behaviors offered by MATSim, we can test the performances of a recently proposed Partially Observable Discrete Event Decision Process (PODEDP) algorithm [18], a policy optimization technique that assumes the environment is only partially observable, and of other state-of-the-art algorithms in a context in which state-transitions are not given *a priori*. Finally, we show that PODEDP can scale and optimize policies for large areas such as an entire city.

The paper is organized as follows. Section II discusses previous works on modeling transportation dynamics using reinforcement and deep learning approaches, also highlighting their limitations. In Section III, we provide an overview of reinforcement learning concepts and the Discrete Event Decision Process (DEDP) [19] algorithm as the basis of PODEDP. In Section IV, we first discuss how we turn MATSim into a reinforcement learning-based environment and then we explore in detail the PODEDP technique. Finally, we briefly introduce the other policy optimization techniques that are benchmarked with PODEDP. In Section V, we outline the experimental setup as well as describe the synthetic and real-world datasets used for the evaluation. The results of the experiments are presented and discussed in Section VI and, finally, in Section VII we propose some conclusions and we suggest possible future directions.

## II. LITERATURE REVIEW

Previous work on optimizing transportation dynamics includes controlling the schedule of traffic lights to reduce traffic jams through dynamic programming and reinforcement learning at the scale of several road intersections and identifying transportation-related policies to optimize transportation network operations through simulation of a city. To the best of our knowledge, the work on optimizing city-scale transportation dynamics based on reinforcement learning and human mobility data is rare, because both the algorithms to cope with complex interaction dynamics and the data are becoming available only in recent years.

In the domain of controlling traffic lights' schedule, the most traditional methods are based on dynamic programming to solve an optimization problem according to precise manually-described assumptions, with *actuated* and *adaptive* methods. The *actuated* methods take into account the provided distribution of the traffic on the roads and set the traffic lights to reduce traffic jams. A significant weakness of models such as [20], [21] is that the optimal time threshold rigidly depends on the volume of the traffic and this parameter has to be set manually. Thus, it is nearly impossible to use such models in modern mega-cities where transportation patterns constantly change, and traffic volume depends on an astonishingly large amount of parameters [22]. The *adaptive* methods, such as Split Cycle Offset Optimisation Technique (SCOOT) and Sydney Coordinated Adaptive Traffic System (SCATS) [23], [24] choose among a set of signal-cycle plans to optimize the assumed traffic distributions. These two models handle traffic congestion inefficiently due to limited adaptability in handling non-recurrent traffic jams in real-time scenarios. Other examples of adaptive methods are Real-Time Hierarchical Optimized Distributed Effective System (RHODES) [25], Optimized Policies for Adaptive Control (OPAC) [26], and Adaptive Control Software Lite (ACS-Lite) [27]. Such models are exponentially complex to deploy them at scale.

Recurrent and non-recurrent big traffic jams reveal the limits of adaptive traffic control methods. In recent years, several works have used a Reinforcement Learning (RL) framework to optimize the management of traffic signals and traffic lights [28], [29] in a data-driven fashion. In RL approaches, one or more agents control the status of the signals according to certain conditions and a reward function. The agent's behavior is usually learned through a substantial number of

simulations to maximize the future expected reward. The key components of an RL algorithm are the *state features*, the *actions*, and the *reward function*. The *state features* include the length of the queue to represent the state of the traffic congestion [30]–[32] and the position of the vehicles with respect to the stop line and its relative speed [29], [33]. The *reward function* for the agent to optimize could be formed from the length of the queue [31], [34], the average delay [28], [35], the cumulative delay [33], [36], and the vehicle staying time [29], or a weighted sum of the length of the queue, the vehicle delays, the average waiting time, the light switcher state (0 if it does not change, 1 otherwise), the number of vehicles and, finally, the total travel time of each car [34].

Using RL algorithms, researchers have recently obtained significant results in adaptive control of traffic signals. Works such as [29], [33], [34], [36] focused on the optimization of single intersections and others took into account multiple crossroads [16], [17], [35], [37]. In [30], a single intersection scenario is used to train an agent and the obtained insights are used to extend this scenario to the management of multiple intersections. Only a couple of these works are extended to a district level: [32] in downtown Toronto and [37] in a restricted area of Barcelona. Regarding the adopted algorithms, most of the mentioned works use either Q-Learning [38], [39] and Deep Q-Learning [40] algorithms to learn to assess the expected reward function of state features, or Policy Gradient [41], [42] algorithms to learn to take actions directly according to the state features. A different method proposed by El-Tantawy *et al.* [32] is based on a Multi-Agent Reinforcement Learning (MARL) framework to coordinate the potentially conflicting goals of the agents.

In the field of transportation policy research [15] and urban planning [43], simulation is a widely-used approach to evaluate transportation policies and identify optimal solutions at the city scale. The utilities [44] to optimize generally involve less traveling time, less uncertainty in arrival time, and less impact of traffic fluctuation on planned activity duration, and the control variables generally involve when to end the current activity and what route to select in the current trip [45], [46]. Experiments in the real world are often costly, dangerous, and infeasible. Simulators provide a solution for evaluating hypotheses and methodologies *in silico* where certain aspects of the behavior faithfully mirror the real world. Transportation simulators [14], [15], [47] generally identify the optimal policy as open-loop control, where the current state of a complex system does not affect the control variables. An open-loop system is simple to implement. Yet it is inaccurate and unreliable when the system is noisy.

Despite all the proposed solutions representing a valuable improvement in traffic optimization, to the best of our knowledge, there is not a solution focused on the optimization of complex systems such as transportation dynamics at a city scale based on high-fidelity simulation [15], big mobility data [48], and reinforcement learning [49]. In this work, we fill the gap by introducing a framework based on the state-of-art transportation simulator MATSim and a reinforcement learning approach.

## III. BACKGROUND

In this section, we provide the background knowledge for MATSim, Reinforcement Learning, Discrete Event Decision Processes (DEDPs), and Partially Observable Discrete Event Decision Processes (PODEDPs).

### A. SIMULATION AND MATSIM

Simulations are extensively used in transportation engineering and policy research as well as in other fields such as urban planning [43], robotics [50], epidemiology [51], gaming [52], and so on. Experiments in the real world are often costly, dangerous, and infeasible. Thus, simulators provide a solution for evaluating hypotheses and methodologies *in silico* where certain aspects of the behavior faithfully mirror the real world.

MATSim [15] is a state-of-the-art large-scale multi-agent transportation simulator: it reproduces a realistic behavior of how people travel and perform activities such as spending time at work, attending schools, shopping, and visiting friends, and how roads with different parameters respond to travel demands. Specifically, the simulator's workflow to identify the typical travel behavior is organized as follows. First of all, the road network is specified in the simulator to match the real world, and initial travel plans (i.e., activities and trips connecting the activities on an individual basis) are created from where people live and work according to census and surveys. Second, the plans for the individuals are executed in the road network through high-fidelity simulations and scored according to their economic values. Next, the simulator re-plans, re-executes, and re-scores the plans for the individuals using a co-evolutionary algorithm until nobody can unilaterally improve their trips. At that point, the system has an equilibrium, and we can inspect the typical behaviors of the individuals.

### B. REINFORCEMENT LEARNING

A reinforcement learning system is delineated by four essential parts: (i) a policy, (ii) a reward function, (iii) a value function, and optionally (iv) a model of the environment [49]. A policy is a mapping between the state perceived by the agent and the actions to take. The reward function determines the goal of the agent and thus the action to take in the immediate. Similarly, the value function of a system determines what is valuable in the long term. The presence of a formal model of the environment discriminates among model-based and model-free RL approaches [49]. More precisely, a model, given a specific state and action, might be used to foretell the next state and the next reward. In other terms, model-based RL approaches can investigate prospective scenarios and situations before they are encountered. On the contrary, model-free methods learn directly from their experiences [49].

Usually, models are employed to mimic the conditions in which the RL algorithm will act and provide the simulated experience. The latter can then be utilized for planning the subsequent actions. In particular, the two ways in which planning can be developed are state-space planning and plan-space planning. In the former, the purpose is to examine all the possible states to determine an optimal policy. In contrast, in the latter, the exploration for an optimal pattern is scrutinized over the space of plans. In some model-based RL settings, an optimal policy is difficult to achieve due to the barriers in modeling large-dimensional state-spaces. This is the case, for example, of transportation systems. Additionally, representing real-world problems as fully observed models is prohibitive in the vast majority of the cases.

### C. DISCRETE EVENT DECISION PROCESS AND PARTIALLY OBSERVABLE DISCRETE EVENT DECISION PROCESS

A Discrete Event Decision Process (DEDP) [19] and a Partially Observable Discrete Event Decision Process (PODEDP) [18] have been proposed for optimal control in transportation systems.

The most significant limitation of the Discrete Event Decision Process (DEDP) [19] is that by considering the full observability of the environment, the algorithm for the policy optimization never performs an information gain check. In particular, DEDP can be seen as a tuple DEDP$\langle S, A, \mathcal{V}, C, P, R, \gamma\rangle$ where $S$ is the state space, $A$ is the action space, $\mathcal{V}$ the set of possible events, $C$ is a mapping between actions and event coefficients, $P$ is the transition kernel, $R$ represents the reward function and, finally, $\gamma \in [0, 1]$ is a discount factor. Mathematically, the transition kernel is represented as the probability of an event to happen given a state and an action $P(s_{t+1}) = p(v_t|s_t, a_t)\delta_{s_{t+1}=s_t+\Delta v}$ and the immediate reward function $R$ is a function of a state at a given time $t$ and corresponds to the sum of the rewards of each component: $R(s_t) = \sum_{\hat{m}=1}^{M} R^{(}\hat{m})_t(s_t^{\hat{m}})$. In this sense, a policy $\pi$ can be defined as a mapping between a state $s_t$ and a distribution of actions $a_t$ parametrized with $\theta$: $\pi p(a_t|s_t; \theta)$. The goal of DEDP is to optimize the policy $\phi$ or, in other terms, maximize the expected future reward as $\theta \leftarrow \text{argmax}_\theta \mathbf{E}_\xi(\sum_t \gamma^t R_t; \theta)$ where $\xi$ is an observed trajectory of the form $\xi = (o_1, a_1, r_1, o_2, a_2, r_2 \ldots, o_T, a_T, r_T)$. The algorithm proposed in [19] (Algorithm 1) optimizes DEDP policy through performing policy evaluation and policy improvement steps repeatedly until convergence:

DEDP policy evaluation through dynamic programming is intractable, due to the exponential size of the state space in the number of state and control variables. In [19], the Bethe entropy approximation [53] is applied to identify a variational lower bound $L(\theta)$ of the log expected future reward. In particular, given a deterministic policy $a_t = \mu(s_t; \theta)$ and the generative model of the (intractable) DEDP process $P(\xi_T; \pi) = p(s_0)\prod_{t=1}^{T}\left(p(v_t \mid s_t; \theta)\delta_{s_{t+1}=s_t+\Delta_{v_t}}\right)$, they introduced a tractable auxiliary process $q(T, m, \xi_T) = q(T, m)q(s_0)\prod_{t=1}^{T} q(s_{t-1,t}, v_{t-1}|T)/\prod_{t=1}^{T-1} q(s_t|T)$ to approximate the assignment of future expected reward among the future states and actions, and solved the variational lower bound with the method of Lagrange multipliers. The result is a forward-backward algorithm to estimate the state statistics $\alpha_t^{(\hat{m})}(s_t^{(\hat{m})})$ and reward statistics $\beta_t^{(\hat{m})}(s_t^{(\hat{m})})$ repeatedly until convergence in Algorithm 1. DEDP Policy improvement is to maximize the log expected future reward lower bound $L(\theta)$, which is constructed from the forward statistics $\alpha_t^{(\hat{m})}(s_t^{(\hat{m})})$ and backward statistics $\beta_t^{(\hat{m})}(s_t^{(\hat{m})})$, and optimized through chain rule.

PODEDP is an approach that aims to optimize a policy starting from a partially observable environment. In general, we can consider it as an extension of DEDP, where the decision-making process is not fully observable. Therefore, we do not need to assume perfect *a priori* knowledge. Moreover, by removing such an assumption, the optimizer is forced to perform an information gain step during the policy optimization process. It leads to the ability to model complex systems such as transportation in a more precise and realistic way. In general, it is known that the majority of the real-world systems are partially observable and, despite the excellent results achieved in [19] using DEDP, we expect to gain better and more realistic results using PODED. In this paper, we integrate PODEDP in an RL environment as a

**Algorithm 1** Optimal Policy of DEDP

**Input:** DEDP$\langle S, A, \mathcal{V}, C, P, R, \gamma\rangle$

**Output:** The optimized parameter $\theta$

**Procedure:** Repeat the following policy evaluation and policy improvement steps until convergence.

1) **Policy evaluation:** Repeat Eq. 1 and 2 to estimate state statistics $\alpha_t^{(\hat{m})}(s_t^{(\hat{m})})$ and reward statistics $\beta_t^{(\hat{m})}(s_t^{(\hat{m})})$ until convergence.

$$\alpha_t^{(\hat{m})}(s_t^{(\hat{m})}) \approx \sum_{s_{t-1}^{(\hat{m})}, v_{t-1}} \alpha_{t-1}^{(\hat{m})}(s_{t-1}^{(\hat{m})}) \cdot p(s_t^{(\hat{m})}, v_{t-1}|s_{t-1}^{(\hat{m})}; \theta). \tag{1}$$

$$\beta_t^{(\hat{m})}(s_t^{(\hat{m})}) = \sum_m q(t, m)\beta_{t|t,m}^{(\hat{m})}(s_t^{(\hat{m})}) + \sum_{s_{t+1}^{(\hat{m})}, v_t} p(s_{t+1}^{(\hat{m})}, v_t|s_t^{(\hat{m})}; \theta)\beta_{t+1}^{(\hat{m})}(s_{t+1}^{(\hat{m})}). \tag{2}$$

2) **Policy improvement:** update $\theta$ with Eq. 3 to maximize the expected future reward function $L(\theta) = \sum_{T,m,\xi_T} q(T, m, \xi_T; \theta^{\text{old}})\log\left(\gamma^T P(\xi_T; \theta)R_T^{(m)}\right)$ with gradient ascent.

$$\frac{\partial L(\theta)}{\partial \theta} = \sum_{t,s_t} \frac{\prod_{\hat{m}} \alpha_t^{(\hat{m})}(s_t^{(\hat{m})}, v_t=v)\beta_t^{(\hat{m})}(s_t^{(\hat{m})}, v_t=v)}{c_v} \frac{\partial c_v}{\partial \theta} - \sum_{t,s_t} \frac{\prod_{\hat{m}} \alpha_t^{(\hat{m})}(s_t^{(\hat{m})}, v_t=\emptyset)\beta_t^{(\hat{m})}(s_t^{(\hat{m})}, v_t=\emptyset)\cdot\prod_m g_v^m(s_t^{(m)})}{1-\sum_{v=1}^{V} c_v\cdot\prod_m g_v^m(s_t^{(m)})} \frac{\partial c_v}{\partial \theta}. \tag{3}$$

policy optimizer and we connect its dynamics to the simulator MATSim described in Section IV-A to evaluate the performances in a realistic city-scale scenario.

## IV. METHODOLOGY

Here, we describe how we integrate a reinforcement learning approach into the MATSim simulator (see Section IV-A). Then, in Section IV-B, we present the Partially Observable Discrete Event Decision Process (PODEDP) algorithm as well as we discuss, in Section IV-C, some additional policy optimization algorithms that can be adopted in the context of transportation systems.

### A. TRANSFORMING MATSIM INTO A REALISTIC REINFORCEMENT LEARNING-BASED ENVIRONMENT

In this work, we transform MATSim, a high-fidelity multi-agent transportation simulator, into a realistic reinforcement learning environment to optimize and evaluate transportation policies. This reinforcement learning environment enables us to assess transportation policies and convince policy-makers by simulating how agents make daily activities and trips with high-fidelity. With the reinforcement learning environment, we develop a model-based reinforcement learning algorithm to approximate MATSim dynamics with a Partially Observed Discrete Event Decision Process and identify the optimal policy through variational inference. The model-based reinforcement learning algorithm is benchmarked against three state-of-the-art algorithms, Policy Gradient, Actor-Critic, and Guided Policy Search, in the proposed reinforcement learning environment on four different scenarios: a fully synthetic one provided by MATSim (SynthTown) and other three real-world datasets involving the cities of Berlin, Santiago de Chile and Dakar. Of the four algorithms, only the model-based reinforcement learning algorithm can converge with a reasonable computational effort. An implementation of the entire framework is available at github.com/LuckysonKhaidem/matsim-code-examples.

One of the challenges we face while translating MATSim into a reinforcement learning environment is keeping track of immediate rewards. This is particularly difficult because the MATSim scoring function generates scores of plans at the end of every iteration. Instead, in our reinforcement learning paradigm, we define immediate rewards as the difference between rewards accumulated between two consecutive actions that an agent takes. To calculate this difference, we need to keep track of rewards that an agent accumulates throughout the course of simulation every minute of the day.

In sum, turning MATSim into a reinforcement learning environment with realistic agents' and roads' behavior, we offer a way to evaluate a transportation dynamic policy in a realistic scenario. In the following, we describe in detail the solution adopted (i.e., Partially Observable Discrete Event Decision Process) to characterize the decision-making process of a transportation system.

### B. REINFORCEMENT LEARNING IN MATSIM WITH A PODEDP

Modeling transportation dynamics on a city-scale represents a formidable hurdle due to the massive dimensions of the state-space and the complexity of the transitions. Furthermore, transportation systems are only partially perceptible and, consequently, it is hard to derive a model that can accurately trace the related dynamics. Another challenge stems from the complex interactions in the system, with the consequence that short-term choices may have massive implications in long-term scenarios. To address those challenges, we built on the PODEDP optimal control algorithm recently proposed by Yang *et al.* [18] and developed a model-based reinforcement learning algorithm. The mathematical notations used in this Section are described in Table 1

**TABLE 1. Summary of the mathematical notation used in this Section with the associated meaning.**

| Symbol | Meaning |
|---|---|
| $S = \{s_1, \dots s_n\}$ | represents the state space |
| $A = \{a_1, \dots a_m\}$ | corresponds to the action space |
| $\mathcal{V} = \{v_1, \dots v_l\}$ | it is the set of the events that may occur |
| $C$ | a mapping function between actions and event coefficients |
| $P$ | the transition kernel: the probability of an event to happen given a state and an action |
| $O$ | the observation function |
| $o_t = (o_t^{(1)}, \cdots, o_t^{(M)}, t)$ | an instance of an observation at time $t$ and location $1 \dots M$ |
| $R$ | the reward function |
| $a_t = \mu(\bullet, \theta)$ | an instance of a deterministic policy |
| $\pi = p(a_t \vert s_t; \theta)$ | an instance of a stochastic policy |
| $\gamma$ | a discount factor |
| $\theta$ | policy parameters |
| $\phi_c$ | model parameter governing state transition dynamics |
| $\phi_R$ | model parameter governing the reward to be received |
| $\mathcal{D}$ | a dataset |
| $\xi$ | an observed trajectory |
| $\alpha, \beta$ | coefficients |

As described by Yang *et al.* [18], a PODEDP is a tuple PODEDP$\langle S, A, \Omega, \mathcal{V}, C, P, O, R, \gamma \rangle$ where $S$ is the state space, $A$ the action space, $\Omega$ the observation space and $O$ is the related observation function. $\mathcal{V}$ is the events set, $C$ is a mapping between actions and event rate coefficients, $P$ corresponds to the transition kernel, $R$ is the reward function and, finally, $\gamma$ is a discount factor that varies between 0 and 1.

Given this definition, the two model parameters are $\phi_c$ governing state transition dynamics and $\phi_R$ governing the immediate reward to be received. The probability for an event $v_t = 1, \dots, V$ to happen conditioned on state $s_t$ and action $a_t$ is $p(v_t \mid s_t, a_t; \phi_c) = h_v(s_t, c_v(a_t; \phi_c)) = c_v(a_t; \phi_c) \prod_{m=1}^{M} g_v^{(m)}(s_t^{(m)})$. In other words, action $a_t$ controls discrete event decision process dynamics through changing the event rate coefficients $\boldsymbol{c} = (c_1, \dots, c_V) = C(a_t; \phi_c)$. The probability of no event is thus $p(v_t = \emptyset \mid s_t, a_t; \phi_p) = 1 - \sum_{v=1}^{V} h_v(s_t, c_v(a_t; \phi_p))$. The immediate reward received by a system at state $s_t$ is $R(s_t; \phi_R) = \sum_{m=1}^{M} R_t^{(m)}(s_t^{(m)}; \phi_R^{(m)})$.

Similarly, the policy parameters are $\theta$. Finally, given an action $a_t = (a_t^{(1)}, \ldots, a_t^{(D)})$, a policy is defined deterministically as $a_t = \mu(s_t; \theta)$ or stochastically as $\pi = p(a_t|s_t; \theta)$.

The idea behind the integration of PODEDP in an RL environment is to start from the model parameters $(\phi_c, \phi_R)$ and the policy parameters $\theta$ and iterates through the following steps until convergence:

1) Execute policy $\mu(\bullet; \theta)$ in the environment. Add the observed trajectory $\xi = (o_1, a_1, r_1, o_2, a_2, r_2 \ldots, o_T, a_T, r_T)$ to the training data set $\mathcal{D} = \mathcal{D} \cup \xi$.
2) Train model parameters $\phi_c$ and $\phi_R$ on dataset $\mathcal{D}$ via Maximum Likelihood Estimation $\phi_c, \phi_R \leftarrow \text{argmax}_{\phi_c,\phi_R} \mathbf{E}_{\mathcal{D}} \log p(\xi; \phi_c, \phi_R)$.
3) Optimize the policy parameter $\theta$ under PODEDP model with parameters $\phi_c$ and $\phi_R$. $\theta \leftarrow \text{argmax}_\theta \mathbf{E}_{\theta,\phi_c,\phi_R} \sum_t \gamma_t R(s_t)$.

Regarding point (1), we optimize city-scale road traffic streams through dispatching carriers to downstream road links in the correct proportion and recommending people regulate the place and duration of their future actions, according to our estimation of traffic jams from the isolated measurements of vehicle trips.

Specifically, the action variables $a_t$ are the chances of choosing a downstream road link after completing the current road link, and the event rate coefficient of leaving or entering buildings. We implement the action variables within MATSim through selecting from the alternative plans identified by the MATSim re-planning module with rejection sampling and using the within-day re-planning interface.

We use PODEDP $\langle S, A, \Omega, \mathcal{V}, C, P, O, R, \gamma \rangle$ to approximate the complex dynamics of MATSim (and the real world) analytically through defining a set of microscopic events. In the PODEDP model, the states $s_t = (s_t^{(1)}, \cdots, s_t^{(M)}, t)$ are the number of vehicles at the $M$ locations (road links and buildings) and the current time $t$. The observations $o_t = (o_t^{(1)}, \cdots, o_t^{(M)}, t)$ are the number of probe vehicles at the $M$ locations and current time $t$, where the probe vehicles are randomly selected and constitutes 10% of the total vehicle population. The movement of vehicles from one location $m_1$ to the next location $m_2$ is represented as an event $p \cdot m_1 \overset{c_{m_1 m_2}}{\rightarrow} p \cdot m_2$, where the event rate coefficient $c_{m_1 m_2}$ is the probability for the vehicle to make the movement. The action variables $a_t$ will change the event rate coefficients to make road usage more efficient. We implement the state transition $p(s_{t+1}, v_t \mid s_t, a_t)$ following the traffic flow diagram to best match MATSim traffic dynamics, including traffic congestion. Then, the reward function $R(s_t)$ is implemented to best approximate the Charypa-Nagel scoring function. Finally, we implement a policy $\mu(s_t; \theta)$ as a neural network with weight $\theta$. The reward function $R(s_t) = \sum_m \beta_{t,\text{perf}}^{(m)} s_t^{(m)} + \beta_{t,\text{trav}}^{(m)} s_t^{(m)}$ emulates the Charypa-Nagel scoring function in transportation research [15] to reward performing the correct activities at facilities and penalize traveling on roads, where $\beta_{t,\text{trav}}^{(m)}$ and $\beta_{t,\text{perf}}^{(m)}$ are the score coefficients. We implement the deterministic policy as a function of states through a neural network $a_t = \mu(s_t) = \mathcal{NN}(s_t; \theta)$ parameterized by policy parameter $\theta$.

Concerning point (2), we used a simplified version of the concepts introduced in [54], [55] to train the two parameters of the model $(\phi_c, \phi_R)$ for each dataset $\mathcal{D}$ introduced in Section V-A. In particular, they propose a way of capturing interactions in social dynamics and of copying exponential growth of state-space with respect to the number of elements in the system. In this sense, it would be unlikely to assess the scenarios utilizing exact inference solutions, and so Xu *et al.* [54] have proposed an approximate inference method. We get inspired by their solution as the complexity of transport dynamics and transportation systems perfectly fit with the obstacle Xu *et al.* [54] solved. Notably, as the state-space is remarkably large and the environment is partially observable, it is better to estimate the parameters of the model using a technique that copes with approximate inference. To succeed, Xu *et al.* [54] suggest to consider $i^{(1)} \ldots i^{(M)}$ as a set of individuals, $v^{(1)} \ldots v^{(T)}$ as a set of events, $y^{(1)} \ldots y^{(T)}$ as a set of observations and, finally, $x^{(1)} \ldots x^{(T)}$ as a sequence of hidden states. In this sense, the likelihood of the sequence is estimated as $P(x^{(1)}, \ldots, x^{(T)}), y^{(1)}, \ldots, y^{(T)}, v^{(1)} \ldots v^{(T)} = \prod_{t=1}^{T} P(x_t, v_t|x_{t-1}) P(y_t|x_t)$. In particular, $P(x_t, v_t|x_{t-1})$ corresponds to the transition kernel. In the policy evaluation step, we use the method introduced by Xu *et al.* [54] to handle the exponential growth of the state-space and to reduce it to a tractable one with a mean-field approximation of the state $q(s_t|T, m) = \prod_{m=1}^{M} q(s_t^{(m)}|T, m)$. In this way, we can define a projected marginal kernel that can be further developed into a backward-forward algorithm. The forward and backward steps are defined as:

$$\alpha_t^{(m)}(x_t^{(m)}) \leftarrow \frac{1}{Z_t} \sum_{x_{t-1}^{(m)}, v_t} \hat{\alpha}_{t-1}^{(m)}(x_{t-1}^{(m)}) \times \hat{P}(x_t^{(m)}, v_t|x_{t-1}^{(m)}) P(y_t^{(m)}|x_t^{(m)}) \quad (4)$$

$$\hat{\beta}_{t-1}^{(m)}(x_{t-1}^{(m)}) \leftarrow \frac{1}{Z_t} \sum_{x_t^{(m)}, v_t} \hat{\beta}_t^{(m)}(x_t^{(m)}) \hat{P}(x_t^{(m)}, v_t|x_{t-1}^{(m)}) P(y_t^{(m)}|x_t^{(m)}) \quad (5)$$

To learn the aforementioned parameters, we maximize the expected log-likelihood

$$\phi_c, \phi_R \leftarrow \text{argmax}_{\phi_c,\phi_R} \mathbf{E}_{\mathcal{D}} \log p(\xi; \phi_c, \phi_R)$$

In our experiments, we use a simplified version of the method proposed to estimate the parameters $\phi_c$ and $\phi_R$.

Finally, for point (3), we need to optimize the policy parameter $\theta$ using the estimated values of $\phi_c$ and $\phi_R$. In other terms, $\theta \leftarrow \text{argmax}_\theta \mathbf{E}_{\theta,\phi_c,\phi_R} \sum_t \gamma_t R(s_t)$. To accomplish this purpose, we execute the algorithm stated in [18]. The policy improvement step performs $\theta \leftarrow \theta + \gamma \frac{\partial \log V^\pi(r)}{\partial \theta}$ where $\gamma$ is the learning rate and the gradient is mathematically described in Equation 3. Given the difficulty of solving a PODEDP problem due to the critical dimension of belief state-space, as recommended in the literature [56], we train

the model in a fully observable environment that can be constructed using DEDP. Adopting such an approach leads to a computationally-simpler solution of the partially observable environment. In different words, we are reducing the optimization of a PODEDP problem to a policy optimization in a Specially designed DEDP (SDEDP). Thanks to the full observability assumed in a DEDP model, we are assured that the optimized value function of a DEDP system is the upper bound of the related PODEDP optimizer. Maximizing the upper bound of the DEDP problem is not enough as we cannot solve the PODEDP model because by computing this bound we are not guaranteeing the performances of the target function. This is where SDEDP plays a crucial role. SDEDP is a tuple SDEDP$\langle S, A, \mathcal{V}, C, P, R, \gamma\rangle$. Given a PODEDP model, SDEDP is the corresponding fully observable model if the two share the same $S, A, \mathcal{V}, C, P, R, \gamma$ and the initial state distribution is the same.

Algorithm 2 summarizes how the local policy associated with the corresponding SDEDP instance can be optimized.

**Algorithm 2** Optimal Policy of the Corresponding SDEDP

**Input:** SDEDP$\langle S, A, \mathcal{V}, C, P, R, \gamma\rangle; \theta$
**Output:** Optimal $\theta$ w.r.t. the stochastic policy
**Procedure:**

1) **Policy evaluation:** Repeat Eq. 4 and 5 until convergence
2) **Policy improvement:** update $\theta$ with Eq. 3

Once done, we are able to estimate the belief state $b_t(s_t) = p(s_t|o_{0:t})$ by inferring the state distribution in the original PODEDP environment from the past observations until the current observation. To obtain an optimized policy in the PODEDP environment, we first apply Algorithm 1 on $\mu^*(s_t)$, we estimate $b_t(s_t)$ as showed in Equation 6 and we generate the optimal policy as indicated in Equation 7.

$$b_t(s_t) = \sum_{\hat{m}=1}^{M} \alpha_t^{\hat{m}}(s_t^{(\hat{m})}) = \propto \sum_{\hat{m}=1}^{M} \sum_{s_{t-1}^{(\hat{m})}, a_{t-1}, v_{t-1}} \alpha_{t-1}^{(\hat{m})}(s_{t-1}^{(\hat{m})}) \times p(s_t^{(\hat{m})}, o_t^{(\hat{m})} a_{t-1}, v_{t-1} | s_{t-1}^{(\hat{m})}; \theta) \quad (6)$$

$$\pi^*(a_t|b_t) = \sum_{s_t} b_t(s_t)\delta_{a_t=\mu^*(s_t)} \quad (7)$$

The steps to follow are further highlighted in Algorithm 3.

As mentioned, the chance of working in a model-based reinforcement learning setting is a unique opportunity to investigate scenarios and learn from simulated experience. Simultaneously, shaping a complex real-world system is challenging, and the risk of training agents on an inadequately designed model is significant. Moreover, model-free RL brings substantial limitations. In such systems, agents cannot carry an explicit plan of how environmental dynamics affect the system, especially in response to an action earlier practiced. Therefore, it would be complicated to collect

**Algorithm 3** Optimization of a PODEDP

**Input:** PODEDP$\langle S, A, \Omega, \mathcal{V}, C, P, O, R, \gamma\rangle$
**Output:** The optimized policy: $\pi^*(a_t \mid b_t)$
**Procedure:**

1) Run Algorithm 2 on $\mu^*(s_t)$
2) Estimate $b_t(s_t)$ with Equation 6
3) Generate the optimal policy as in Equation 7

insights that can be applied to real-world situations. The solution specified in this Section is a model that embodies a good compromise between the ascertainment that in a model-based environment we can gather valuable insights and the admission that a world may be partially observable. In this sense, we affirm that by using such a model, we can capture complex real-world dynamics more accurately and succinctly with respect to other models.

Despite the good results obtained in model-based scenarios that assume the full observability of the environment ( [19]), by removing such strong assumptions ( [18]) we can overtake the results of methods such as DEDP, Guided Policy Search (GPS) [57], Policy Gradient (PG) [58] and Actor-Critic (AC) [59]. The outcomes are discussed and investigated in Section VI.

### C. POLICY OPTIMIZATION TECHNIQUES

In this Section, we focus on three policy optimization algorithms. In particular, we analyze two simulation-based algorithms, namely Policy Gradient (PG) [58] and Actor-Critic (AC) [59], and an analytical method, Guided Policy Search (GPS) [57]. The main difference between simulation and analytical methods is that the former category sample the states and the actions to reproduce population flows while the latter approximate transition dynamics using differential equations and solve local policies. All the mentioned techniques implement the policy as a neural network considering the historical inputs as observations. The performances of the aforementioned state-of-art algorithms are compared with our proposed approach based on PODEDP [18] in Section VI.

Regarding PG [58], the authors have highlighted that in many works, the loss function, that should be optimized to solve a problem successfully, consists of computing an expectation over a set of variables that may be part of a probabilistic environment and thus gradient-based algorithms cannot be used. To overtake this issue, Schulman *et al.* [58] show how an unbiased estimator of the loss function's gradient can be derived starting from a stochastic computation graph: a directed acyclic graph that encodes the dependency structure of the computation to be performed. Four types of nodes can be found in the graph: (i) input nodes containing the parameters we want to compute, the derivative of (ii) deterministic nodes that correspond to a deterministic function we want to calculate with respect to the parents, (iii) stochastic nodes that specify a random variable through a conditional distribution on their parents, and finally (iv) constant nodes, namely a subset of deterministic nodes corresponding to real

numbers. Schulman *et al.* [58] derive a gradient estimator for an expected sum of costs in a stochastic computation graph and they show that this gradient estimator can be computed efficiently. In their paper, the authors propose two ways to calculate the gradient. One way is to use the backpropagation algorithm with one of the surrogate loss functions. As an alternative, the algorithm proposed in [58] can be used. Another algorithm for policy optimization that we compare with PODEDP is proposed by Lillicrap *et al.* [59]. In this work, they introduce an Actor-Critic model-free algorithm that also works in continuous action spaces. Q-learning cannot be applied to continuous action space due to the computational complexity of finding a greedy optimized policy. Thus, the authors have proposed an Actor-Critic algorithm based on a deterministic policy gradient [60].

An additional solution to policy optimization is described in [57]. There are many guided policies that can be used to solve non-linear problems and Montogomery and Levine [57] propose a new Guided Policy Search (GPS) algorithm providing two main contributions. First of all, their algorithm guarantees convergence in simplified convex and linear settings. Moreover, they show that, in the more general non-linear setting, the error in the projection step can be bounded.

## V. EXPERIMENTAL SETUP AND DATA

In this Section, we present the structure of the experiments we perform as well as the synthetic and real-world datasets used for evaluating our approach. Overall, our purpose is to compute an optimal policy to manage transportation dynamics in order to (i) minimize the time cars spend on the roads, (ii) arrive at a specific Point of Interest (POI) on time, and (iii) staying at a given POI for a sufficient amount of time. Some of the features needed to run the PODEDP optimizer are taken from the datasets outlined in Section V-A, while others are measured from MATSim. As previously said, a PODEDP can be seen as a tuple PODEDP$\langle S, A, \Omega, \mathcal{V}, C, P, O, R, \gamma\rangle$ and the mapping done to run our experiments is as follows. First of all, the set of states $S$ is composed by elements such as $s_t = (s_t^{(1)}, \ldots, s_t^{(M)}, t)$ representing the cardinality (number of vehicles) at $M$ locations at a given instant $t$. Moreover, $O$ is an observation function and, as far as most of the real-world complex systems are partially observable, we may want to observe only a subset of these vehicles. In our experiment, we randomly select the 10% of the elements as probe vehicles ($\Lambda$). In this sense, we can model $O$ as a set of items $o_t = (o_t^{(1)}, \ldots, o_t^{(M)}, t)$ to delegate each $\lambda \in \Lambda$ at the $M$ location at a specific time $t$. $\Omega$ is the related observation space and $o_t$ is the observation received at time $t$.

Regarding the events, each event $v$ has the form $p{\cdot}m_1 \overset{c_{m_1 m_2}}{\rightarrow} p{\cdot}m_2$ and highlights how an element $p$ moves from $m_1$ to $m_2$ and each $c = (c_1, \ldots . c_V) \in C$ is $P(s_{t+1}, v_t | s_t, a_t)$. In other terms, $c_{m_1,m_2}$ is the probability of $p$ of moving from $m_1$ to $m_2$ at $t$. Similarly, each action $a_t$ can be represented as the event rate coefficient of (i) choosing the downstream link, (ii) leaving a POI, and (iii) entering a POI and staying there for a sufficient time period. Concerning the traffic dynamics, we decide to follow the traffic flow diagram proposed by MATSim while the reward function $R(s_t) = \Sigma m \beta_{t,\text{perf}}^{(m)} s_t^{(m)} + \beta_{t,\text{trav}}^{(m)} s_t^{(m)}$ emulates the Charypa-Nagel scoring function illustrated in [15]. Finally, we implement the policy as $p(a_t|b_t;\theta) = \Sigma_{s_t} b_t(s_t) \delta_{a_t = \mu(s_t;\theta)}$ where $\mu(s_t;\theta)$ is a neural network with $\theta$ as weights.

A model-based reinforcement learning algorithm is data-efficient. To demonstrate that, we compare our proposed approach against other analytical and simulation methods. For the simulation method, we focus on a Policy Gradient (PG) [58] and an Actor-Critic (AC) [59] that sample the actions and next states to reproduce the population flow. For the analytical method, we focus on a Guided Policy Search (GPS) [57] that approximates the transition dynamics with differential equations and solves the local policies analytically. All these algorithms implement the policy as a neural network with the historical inputs as observations. We run each algorithm on each scenario for 10 times and draw the average and standard deviation of the achieved utilities across training epochs.

### A. DATA

We evaluate the performance of our proposed framework on four datasets of human mobility: (i) SynthTown, (ii) Berlin, (iii) Santiago de Chile, and (iv) Dakar. A summary of the relevant characteristic of each dataset are exposed in Table 2

The SynthTown dataset contains a synthesized network of one home location, one work location, and 23 single-direction road links. Hence, it represents the synthesized travel demand of 50 agents going to work in the morning and returning home in the afternoon [15]. We uniformly sample 10% of these agents as probe ones who are observable on link 1 and link 20. The optimization problem has the goal of maximizing the total utility over all agents through setting the rate for an agent to start the home-to-work and the work-to-home trips, and distributing agents to the available downstream links according to the time of day and the numbers of probe agents on link 1 and link 20. If each agent maximizes his/her utility greedily, traffic congestion will happen and the overall utility will be sub-optimal. The numbers of probe agents on link 1 and link 20, on the other hand, provide information for the controller to optimize agent behavior. This centralized control is what happens when a transportation agency and a web-mapping service-provider control traffic signals, provide traffic information, and diversify the routes to optimize road network operation.

The Berlin dataset is comprised of a network of approximately 2400 single-direction road links, derived from OpenStreetMap, in the metropolitan area of Berlin bounded by the Berlin beltway. Moreover, this dataset contains the trips of 17 thousand synthesized vehicles representing the travel behaviors of three million inhabitants [61]. The daily trips in the Berlin dataset were synthesized from (i) the commuter data provided by the German Federal Employment Agency containing the home and workplace municipalities of the working population, (ii) an activity-based demand model [64]

**TABLE 2.** A synthesis of the datasets used in the experiment and their main characteristics.

| Reference | Dataset | Road Network Source | # Links | # Agents / Vehicles |
|---|---|---|---|---|
| [15] | **SynthTown** | MATSim | 23 | 50 |
| [61] | **Berlin** | OpenStreetMap | 2400 | 17.000 |
| [62] | **Santiago de Chile** | OpenStreetMap | 6000 | 65.000 |
| [63] | **Dakar** | OpenStreetMap | 8000 | 12.000 |

to sample a sequence of activities (staying at home, working, attending schools, shopping, going to the restaurant, etc.) and the corresponding travels throughout a day, (iii) a physical simulation [15] to repeatedly modify the sampled activity-travel sequences to match the capability of the transportation network, and finally (iv) a Bayesian sampling [65] to match the daily activity-travel sequences from the previous step with hourly traffic count values from over 300 count stations. The synthesized daily trips have been validated based on extensive, regularly-conducted travel surveys and constitute a quality representation of road transport demand.

The third dataset, the Santiago de Chile dataset, comprises a network of 6000 single-direction links derived from OpenStreetMap, and of the trips of 65 thousand synthesized vehicles representing the travel behaviors of six million inhabitants in cars, walking and public transportation modals [62]. The daily trips in the Santiago de Chile dataset were initialized from cloning the sequences of activities (starting time and duration of time spent at home, work, school, doing shopping, doing leisure activities, doing visits, and at health facilities) and travel mode of 60 thousand individuals (from 18 thousand households) from publicly-accessible travel diaries. These sequences of activities are modified through physical simulation and a co-evolutionary algorithm (using MATSim) to maximize the overall utility of the system. The resulting daily trips are compatible with travel modals' distributions and observed traffic counts at count stations.

Finally, the Dakar dataset contains a network of 8000 single-direction road links derived from OpenStreetMap and 12 thousand real-world vehicle trips derived from the ''Data for Development (D4D)'' challenge datasets based on the Call Detail Records (CDR) of over 9 million Sonatel customers in Senegal through the year 2013 [63]. A Call Detail Record is a data record that details a telecommunication transaction to generate phone bills and has been used widely in academic research for modeling human mobility patterns [48]. More precisely, the D4D-Senegal datasets contain hourly site-to-site voice/SMS traffic among 1666 sites (dataset 1), mobility of 300 thousand users randomly sampled every 2 weeks at site level (dataset 2), and the mobility of 150 thousand randomly-sampled users for one year at the level of 123 arrondissements (dataset 3). From dataset 2, we identify the home and work/school locations of each user as randomly picked locations from the most appeared sites in the periods 7am - 7pm and 7pm - 7am, respectively. Then, we sample an activity-trip sequence for each user to match her/his sequence of mobility records from a Markov chain model describing how s/he performed various activities (staying at home, staying at work, attending school, shopping, etc).

In MATSim, immediate rewards and penalties are assigned to an agent according to its current location. For example, performing an activity results in decreasing immediate reward over time (diminishing returns), while traveling results in a constant immediate penalty over time depending on the transportation mode and an upfront penalty at the beginning of a trip. Early arrival and departure result in no reward or penalty. Additionally, traffic dynamics in MATSim are modeled mesoscopically as a queuing network. A road link is characterized with maximum speed, maximum flow and maximum capacity. When traffic flow (i.e., number of vehicles moving out of the road link per unit time) is less than the maximum traffic flow, vehicles move out of the link at maximum speed. Otherwise, vehicles are queued up in the road link until all cars in front of them move out the link at maximum traffic flow. When the number of cars in the link reaches maximum capacity, no other vehicles are allowed to move into the link. Both the utility function and the dynamics reflect the complex behavior in real world road traffic.

## VI. RESULTS

In this Section, we compare the performance of Guided Policy Search, Actor-Critic, Policy Gradient, and model-based reinforcement learning with the Partially Observed Discrete Event Decision Process algorithm on the four scenarios described in Section V-A. Since the performance loss of a model-based reinforcement learning algorithm comes from both the inaccuracy of the model representing the real world and the approximation to the optimal solution within the model, we also compare the PODEDP dynamics and the MATSim dynamics it represents.

### A. COMPARING GPS, AC, PG, AND PODEDP MODEL-BASED REINFORCEMENT LEARNING

In a reinforcement learning environment, the final aim is to optimize a policy by maximizing the rewards. In Figure 1, we show the learning curves of PODEDP (in grey), GPS (in red), AC (in green), and PG (in blue) when we optimize transportation dynamics in the SynthTown, Berlin, Santiago, and Dakar scenarios respectively. Moreover, in Figure 1 we highlight the variance of the algorithms, which is represented by the colored area around the curves.

From the panels, it emerges that the PODEDP-based reinforcement learning algorithm quickly achieves higher utilities and with less variance. This is because the algorithm uses the data gathered from executing the learned PODEDP policy to refine the PODEDP model with supervised learning. Then, it uses the refined PODEDP model to improve the policy with variational inference, thus achieving data efficiency. At the same time, we have not been able to bring GPS, AC and PG to

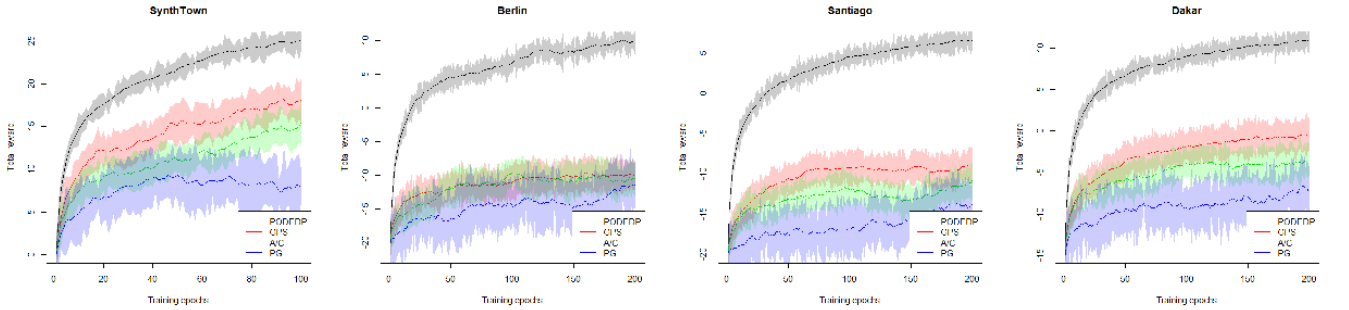


**FIGURE 1.** The total rewards achieved by the algorithms over the training epochs. In the figures, we also highlight the models' variances, represented by the colored areas around the curves. PODEDP (in grey) performs better than the other models in all the scenarios while the analytical solution, represented by GPS (in red), achieves more significant rewards in all the cases with respect to the simulation-based techniques (AC in green and PG in blue).

convergence with a reasonable computational effort. The analytical solution (GPS algorithm) obtains better results with respect to the simulation-based solution. Only in the Berlin scenario, the performances of GPS are close to the one of AC. In general, in a road traffic network, high rewards require the individuals to perform the correct activities (staying at home, working, and so on) at the right time, and to spend less time on roads. Modeling the complex system transition dynamics based on a Markov Decision Process analytically, using Taylor approximation, introduces modeling errors, which is the case of the GPS policy optimization techniques. The PG algorithm uses a similar technique based on a Monte Carlo integration. In this sense when this algorithm deal with high-dimensional state spaces, its variance drastically increase and a small perturbation of policy may result in significant changes to the immediate reward and value in later steps, which makes it difficult to estimate the correct value from sampled trajectories, and as a result difficult to compute the correct value gradient. The AC algorithm also faces the same problem.

To illustrate how the learned behavior of individual agents causes the overall performance differences, we present the average number of vehicles of ten runs at each location of SynthTown for each algorithm using the learned policy (Figure 2). As shown in this Figure, the PODEDP leads to the smallest amount of vehicles on roads, the largest amount of vehicles at work during working hours (9am - 5pm), and the largest amount of vehicles at home during rest hours (other hours), which indicates that the learned policy of PODEDP algorithm best satisfies the needs. By combining the accurate modeling of PODEDP with a tractable solution using variational inference, our method achieves the best performance. For other analytical techniques, GPS introduces modeling error when approximate the state transitions with differential equations. PG and AC instead introduce a high variance in sampling.

To further highlight the performances of the algorithms under analysis, we identify some key performance metrics (Table 3). In particular, we measure the average time vehicles spend on the roads, the number of cars at workplace locations in the working hours (9am - 5pm) and vehicles at

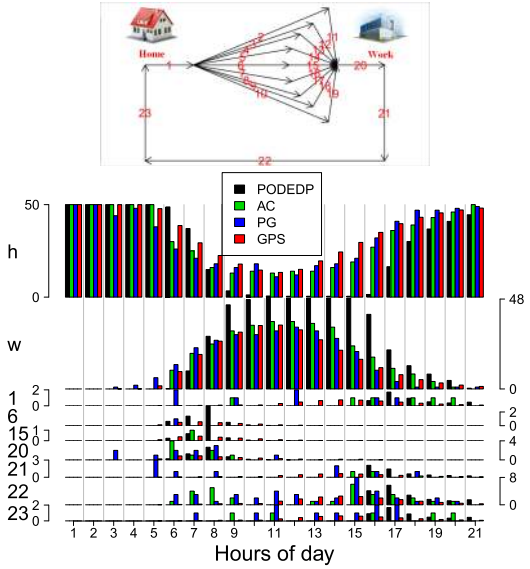


**FIGURE 2.** Average number of vehicles by hour of a day at home, work, and road links with trained policies in the SynthTown scenario (network on the top). PODEDP model-based reinforcement learning achieves least travel time and best on-time arrival.

home locations during non-working hours. These values are collected for each algorithm in ten runs and using the policy that is learned at that point. Such metrics can be used as indicators of how well a learned policy performs. Given the higher rewards obtained by the PODEDP techniques, it is the algorithm that better minimizes the average times of vehicles on the roads and maximizes the number of vehicles both at the workplaces in working hours and at home locations in the other periods.

### *B. COMPARISON OF MATSim AND PODEDP DYNAMICS*

To emphasize the PODEDP solution's correctness, we compare its traffic dynamics with the ones of MATSim. Specifically, we run a 24-hour MATSim simulation of the Berlin environment by executing a predefined set of plans. We collect the location and reward received for each agent in the simulation at every minute of the simulation time. We also record the same metrics for the PODEDP solution that has been run using the learned policy for the Berlin environment.

**TABLE 3.** Performance comparison of algorithms in the following metrics: average time (minutes) on the road per vehicle (TOR), average number of cars on the road per unit time (minutes) (VOR), and average number of vehicles at work during working hours (VAW).

| Dataset | SynthTown | | | Berlin | | | Santiago | | | Dakar | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metrics | **TOR** | **VOR** | **VAW** | **TOR** | **VOR** | **VAW** | **TOR** | **VOR** | **VAW** | **TOR** | **VOR** | **VAW** |
| PODEDP | **25** | **2** | **16** | **80** | **556** | **927** | **45** | **400** | **755** | **65** | **252** | **537** |
| AC | 35 | 3 | 13 | 679 | 4288 | 203 | 468 | 2311 | 295 | 159 | 452 | 234 |
| PG | 56 | 4 | 12 | 793 | 4907 | 179 | 503 | 1140 | 245 | 219 | 586 | 245 |
| GPS | 27 | 2 | 11 | 498 | 3318 | 123 | 397 | 1952 | 352 | 170 | 476 | 266 |

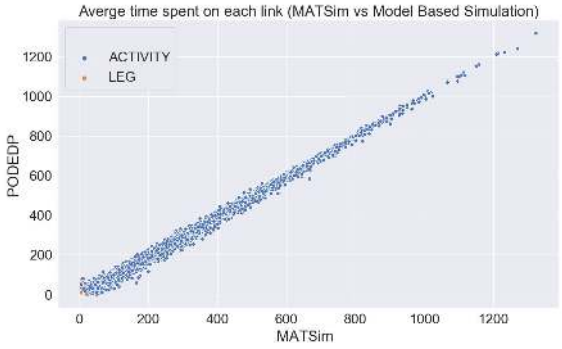


**FIGURE 3.** A comparison of the average time in minutes spent on each link on MATSim (x-axis) and PODEDP (y-axis). The average time is related to activities (e.g., stay in a building for working purposes) highlighted in blue and legs (time spent on streets) depicted with orange dots. The linearity of the distribution highlight that PODEDP and MATSim have similar temporal dynamics.

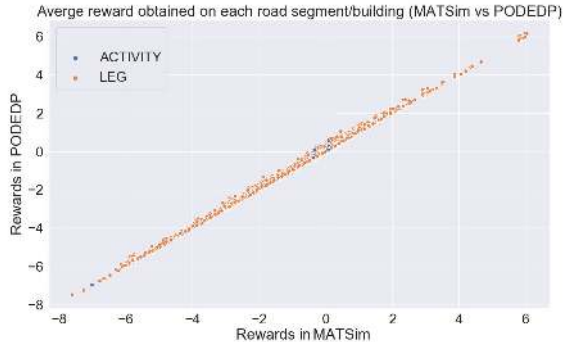


**FIGURE 4.** A comparison between the rewards of MATSim (x-axis) and the ones of PODEDP (y-axis). Also in this case the linear relation spotlight similar dynamics (both for the activities in blue and legs in orange) between the simulator and the proposed policy optimization algorithm.

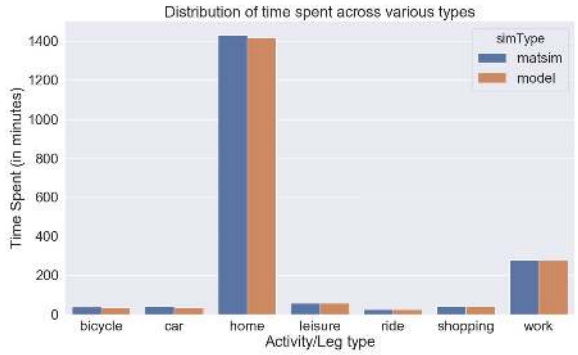


**FIGURE 5.** The distribution of minutes spent across each activity/leg further confirm the similarities between MATSim (in blue) and PODEDP (in orange). Moreover, the time spent on trips is lower in the PODEDP case.

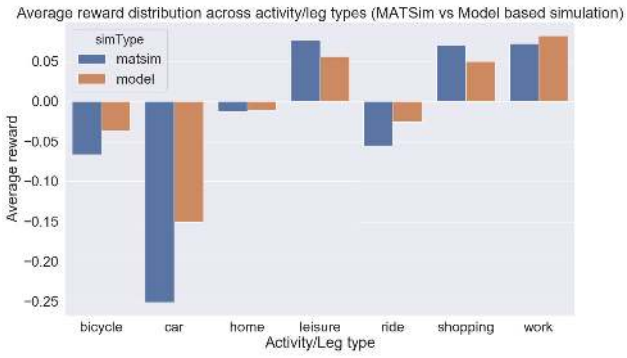


**FIGURE 6.** Regarding the distribution of the average rewards across activities/legs, we show that when dealing with legs (negative rewards) PODEDP -in orange- performs much better than MATSim -in blue- while we obtain similar results in the activity contexts.

Using the data obtained from both the experimental runs, we may draw various insights to shed light on how similar the traffic and reward dynamics of the two simulations are. Since MATSim provides a realistic simulation of the actual traffic dynamics, we need to ensure that our solution is effective in an environment whose dynamics are close to the ones of MATSim's.

Figure 3 compares the average time spent on each link between the two simulations. For each link (i.e., road segment/building), we calculate the average time spent in minutes and then sample 10,000 links and plot their corresponding average duration. Each data point represents a link where its x-coordinate represents the average time spent in MATSim and its y-coordinate represents the average time spent in the PODEDP run. The strong diagonal implies strong similarities in the temporal dynamics of the two simulations.

Similarly, Figure 4 compares the average reward received on each link for both the simulations. The average reward values are calculated in the same manner as we have done for the previous plots. Even in this case, we can observe a substantial similarity in the reward dynamics, as implied by the plot's diagonality. We can also observe from the plot that the corresponding rewards for PODEDP tend to be slightly higher than the ones of MATSim's on some links. Figure 5 compares the average time spent in minutes on different modes of transport (leg) and activity. This plot digs deeper into how average time spent is distributed across different types of legs and activities. We can observe that for the two simulations, the corresponding time spent is almost the same. Note that for PODEDP, the overall time spent on trips is slightly less than that one of its MATSim counterpart. Similarly, Figure 6 compares the average reward received by the agents of the simulations across various activities/legs. As we can see for the legs, where MATSim awards negative rewards, the PODEDP model performs significantly better than the MATSim model. However, for the activities, rewards

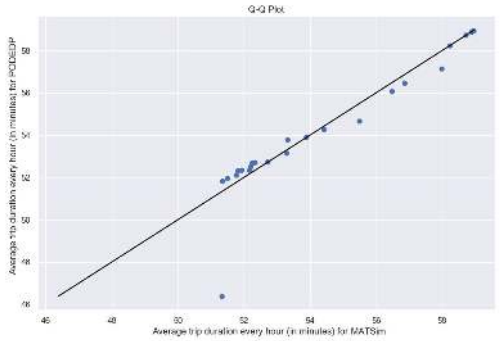


**FIGURE 7.** A quantile-quantile plot to compare the average trip duration for both working and non-working hours (in minutes). On the x-axis, we have the minutes taken for a trip in MATSim and on the y-axis the performance of time spent with PODEDP. In general, PODEDP performs slightly better than MATSim, meaning that, after a PODEDP policy optimization, travelers spend less time on the roads.

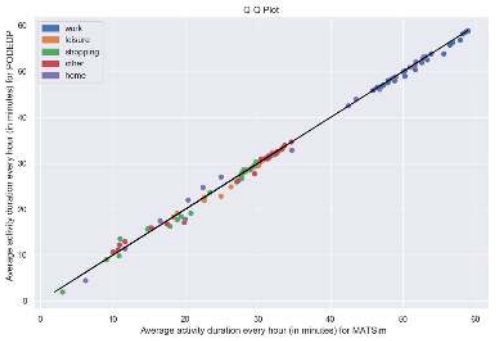


**FIGURE 8.** A quantile-quantile plot regarding the duration of activities every hour (in minutes). In both MATSim (x-axis) and PODEDP (y-axis), all the activities such as work (in blue), leisure (in orange), shopping (in green) and other activities (in red) have similar performances. The only exception is the "stay home" activity (in purple).

earned by the PODEDP based model do not surpass but are very close to the MATSim based model. Based on rewards, we can say that PODEDP model, if not better, is at par with the MATSim based model.

Figure 7 shows the quantile-quantile plot of the average trip duration, for every hour and for both models. As we can see the average time taken by the PODEDP based model is slightly less than the one taken by the MATSim model. This reinforces what we have seen in Figure 3, where the average time spent by the PODEDP model over activities is slightly less than the one by the MATSim model.

Finally, Figure 8 shows the quantile-quantile plot of the average activity duration, for each hour, and for both models. Here we observe that for all the activities, with the exception of staying at home, the two models spend almost the same time. This again confirms our observations from Figure 3. Both models spend the same average amount of time on all activities except home, where the PODEDP based model spends slightly less time. This again shows that our PODEDP based model is comparable to MATSim in terms of performance.

## VII. DISCUSSION AND CONCLUSION

Redesigning and modernizing urban mobility has a pivotal role in our metropolitan landscapes. Yet there is not a city-scale solution to optimize transportation dynamics based on big mobility data and reinforcement learning to the best of our knowledge. In this work, we transformed MATSim, a high-fidelity multi-agent transportation simulator, into a realistic reinforcement learning environment for optimizing and evaluating transportation policies. This reinforcement learning environment enables us to assess transportation policies and convince policy-makers by simulating how agents make daily activities and trips with high-fidelity. With the reinforcement learning environment, we developed a model-based reinforcement learning algorithm to approximate MATSim dynamics with a partially observed discrete-event decision process, and to identify the optimal policy through variational inference. The model-based reinforcement learning algorithm is benchmarked against three state-of-the-art algorithms, Policy Gradient, Actor-Critic, and Guided Policy Search in the proposed reinforcement learning environment on four different scenarios: a fully synthetic one provided by MATSim (SynthTown) and other three real-world datasets involving the cities of Berlin, Santiago de Chile, and Dakar. Of the four algorithms, only the model-based reinforcement learning algorithm can converge with a reasonable computational effort. A detailed comparison between PODEDP dynamics and the corresponding MATSim dynamics using the Berlin scenario demonstrates that PODEDP indeed captures complex system dynamics well. Thus model-based reinforcement learning is sample efficient with the use of variational inference to search the solution space efficiently.

While we have evaluated the proposed model-based reinforcement learning approach in a state-of-the-art multi-agent transportation simulator that emulates real-world road network dynamics highly realistically, we did not evaluate it in the real world. This reflects a limitation of the evaluation. But such restriction is typical in complex systems and policy research because experiments in the real world are often costly, dangerous, and infeasible. Transforming state-of-the-art simulators into a realistic reinforcement learning environment for policy optimization and evaluation, and applying modern reinforcement learning techniques to learn complex systems control from big data represent a new direction in introducing human mobility data and reinforcement learning into policy research. Through the integration of state-of-the-art simulation models and big data with machine learning, we can effectively turn the real world into a living lab, where theories are evaluated on the data commons, results are quantifiable and approaches are repeatable. In the future, we expect to see applications of semantically richer models such as multi-agent reinforcement learning and mean-field game, additional data sources of heterogeneous nature, and eventually real-world evaluations.

## ACKNOWLEDGMENT

*(Luckyson Khaidem, Massimiliano Luca, Fan Yang, Ankit Anand, Bruno Lepri, and Wen Dong contributed equally to this work.)*

## REFERENCES


[1] D. Li, J. Ma, T. Cheng, J. L. van Genderen, and Z. Shao, "Challenges and opportunities for the development of MEGACITIES," *Int. J. Digit. Earth*, vol. 12, no. 12, pp. 1382–1395, Dec. 2019.

[2] B. D. Taylor, "The geography of urban transportation finance," in *The Geography of Urban Transportation*, vol. 3. New York, NY, USA: The Guilford Press, 2004, pp. 294–331.

[3] H. J. Shatz, K. E. Kitchens, and S. Rosenbloom, *Highway Infrastructure Economy: Implications for Federal Policy*. Santa Monica, CA, USA: Rand Corporation, 2011.

[4] J. B. Bump, S. K. Reddiar, and A. Soucat, "When do governments support common goods for health? Four cases on surveillance, traffic congestion, road safety, and air pollution," *Health Syst. Reform*, vol. 5, no. 4, pp. 293–306, Oct. 2019.

[5] N. Zhong, J. Cao, and Y. Wang, "Traffic congestion, ambient air pollution, and health: Evidence from driving restrictions in beijing," *J. Assoc. Environ. Resource Economists*, vol. 4, no. 3, pp. 821–856, Sep. 2017.

[6] W. J. Requia, C. D. Higgins, M. D. Adams, M. Mohamed, and P. Koutrakis, "The health impacts of weekday traffic: A health risk assessment of $PM_{2.5}$ emissions during congested periods," *Environ. Int.*, vol. 111, pp. 164–176, Feb. 2018.

[7] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2010, Nov. 2010. [Online]. Available: https://www.microsoft.com/en-us/research/publication/t-drive-driving-directions-based-on-taxi-trajectories/

[8] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on GPS data," in *Proc. 10th Int. Conf. Ubiquitous Comput. (UbiComp)*, 2008, pp. 312–321.

[9] L. Pappalardo, S. Rinzivillo, Z. Qu, D. Pedreschi, and F. Giannotti, "Understanding the patterns of car travel," *Eur. Phys. J. Special Topics*, vol. 215, no. 1, pp. 61–73, Jan. 2013.

[10] J. Yoon, B. Noble, and M. Liu, "Surface street traffic estimation," in *Proc. 5th Int. Conf. Mobile Syst., Appl. Services (MobiSys)*, New York, NY, USA, 2007, p. 220, doi: 10.1145/1247660.1247686.

[11] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 871–882, Jun. 2013.

[12] L. Huang, Y. Yang, X. Zhao, C. Ma, and H. Gao, "Sparse data-based urban road travel speed prediction using probabilistic principal component analysis," *IEEE Access*, vol. 6, pp. 44022–44035, 2018.

[13] Y. Huang, L. Qian, A. Feng, N. Yu, and Y. Wu, "Short-term traffic prediction by two-level data driven model in 5G-enabled edge computing networks," *IEEE Access*, vol. 7, pp. 123981–123991, 2019.

[14] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO-Simulation of Urban MObility," *Int. J. Adv. Syst. Meas.*, vol. 5, nos. 3–4, pp. 128–138, 2012.

[15] A. Horni, K. Nagel, and K. W. Axhausen, *The Multi-Agent Transport Simulation MATSim*. London, U.K.: Ubiquity Press, 2016.

[16] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," 2016, *arXiv:1611.01142*. [Online]. Available: http://arxiv.org/abs/1611.01142

[17] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, 2008, pp. 656–671.

[18] F. Yang, B. Lepri, and W. Dong, "Optimal control in partially observable complex social systems," in *Proc. 18th Int. Conf. Auto. Agents MultiAgent Syst.*, 2020, pp. 1557–1565.

[19] F. Yang, B. Liu, and W. Dong, "Optimal control of complex systems through variational inference with a discrete event decision process," in *Proc. 18th Int. Conf. Auto. Agents MultiAgent Syst.* Richland, SC, USA: International Foundation for Autonomous Agents and Multiagent Systems, 2019, p. 296–304.

[20] M. Fellendorf, "VISSIM: A microscopic simulation tool to evaluate actuated signal control including bus priority," in *Proc. 64th Inst. Transp. Eng. Annu. Meeting*, vol. 32. Dallas, TX, USA: Springer, 1994, pp. 1–9.

[21] P. Mirchandani and L. Head, "A real-time traffic signal control system: Architecture, algorithms, and analysis," *Transp. Res. C, Emerg. Technol.*, vol. 9, no. 6, pp. 415–432, Dec. 2001.

[22] T. F. Golob and W. W. Recker, "Relationships among urban freeway accidents, traffic flow, weather, and lighting conditions," *J. Transp. Eng.*, vol. 129, no. 4, pp. 342–353, Jul. 2003.

[23] R. Akcelik, "Traffic signals: Capacity and timing analysis," ARRB, Melbourne, VIC, Australia, Res. Rep. 123, 1981.

[24] A. Stevanovic, C. Kergaye, and P. T. Martin, "Scoot and scats: A closer look into their operations," in *Proc. 88th Annu. Meeting Transp. Res. Board*, Washington, DC, USA, 2009.

[25] P. Mirchandani and F.-Y. Wang, "RHODES to intelligent transportation systems," *IEEE Intell. Syst.*, vol. 20, no. 1, p. 10–15, Jan./Feb. 2005, doi: 10.1109/MIS.2005.15.

[26] N. H. Gartner, P. J. Tarnoff, and C. M. Andrews, "Evaluation of optimized policies for adaptive control strategy," *Transp. Res. Rec.*, no. 1324, pp. 105–114, 1991.

[27] D. Gettman, S. Shelby, K. L. Head, and R. Ghaman, "Overview of the ACS-lite adaptive control system," in *Proc. 13th World Congr. Intell. Transp. Syst. Services (ITS)*, 2006, pp. 2976–2988.

[28] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *Proc. Learn., Inference Control Multi-Agent Syst. (2016)*, 2016.

[29] J. Gao, Y. Shen, J. Liu, M. Ito, and N. Shiratori, "Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network," 2017, *arXiv:1705.02755*. [Online]. Available: http://arxiv.org/abs/1705.02755

[30] B. Abdulhai, R. Pringle, and G. J. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," *J. Transp. Eng.*, vol. 129, no. 3, pp. 278–285, May 2003.

[31] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA J. Automatica Sinica*, vol. 3, no. 3, pp. 247–254, Jul. 2016.

[32] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown toronto," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1140–1150, Sep. 2013.

[33] X. Liang, X. Du, G. Wang, and Z. Han, "Deep reinforcement learning for traffic light control in vehicular networks," 2018, *arXiv:1803.11115*. [Online]. Available: http://arxiv.org/abs/1803.11115

[34] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: A reinforcement learning approach for intelligent traffic light control," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2496–2505.

[35] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intell. Transp. Syst.*, vol. 4, no. 2, pp. 128–135, 2010.

[36] S. S. Mousavi, M. Schukat, and E. Howley, "Traffic light control using deep policy-gradient and value-function-based reinforcement learning," *IET Intell. Transp. Syst.*, vol. 11, no. 7, pp. 417–423, Sep. 2017.

[37] N. Casas, "Deep deterministic policy gradient for urban traffic light control," 2017, *arXiv:1703.09035*. [Online]. Available: http://arxiv.org/abs/1703.09035

[38] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College Cambridge, Univ. Cambridge, Cambridge, U.K., 1989.

[39] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.

[40] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*. [Online]. Available: http://arxiv.org/abs/1312.5602

[41] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 1057–1063.

[42] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Proc. Adv. Neural Inf. Process.Syst.*, 2000, pp. 1008–1014.

[43] P. Waddell, "UrbanSim: Modeling urban development for land use, transportation, and environmental planning," *J. Amer. Planning Assoc.*, vol. 68, no. 3, pp. 297–314, Sep. 2002.

[44] S. Russel and P. Norvig, *Artificial Intelligence: A Modern Approach*. London, U.K.: Pearson, 2013.

[45] M. E. Ben-Akiva, S. R. Lerman, and S. R. Lerman, *Discrete choice Analysis: Theory and Application to Travel Demand*. Cambridge, MA, USA: MIT Press, 1985, vol. 9.

[46] D. Charypar and K. Nagel, "Generating complete all-day activity plans with genetic algorithms," *Transportation*, vol. 32, no. 4, pp. 369–397, Jul. 2005.

[47] L. Smith, R. Beckman, and K. Baggerly, "Transims: Transportation analysis and simulation system," Los Alamos Nat. Lab., Los Alamos, NM, USA, Tech. Rep., 1995.
[48] V. D. Blondel, A. Decuyper, and G. Krings, "A survey of results on mobile phone datasets analysis," 2015, *arXiv:1502.03406*. [Online]. Available: http://arxiv.org/abs/1502.03406
[49] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html
[50] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 5026–5033.
[51] S. Eubank, H. Guclu, V. S. Anil Kumar, M. V. Marathe, A. Srinivasan, Z. Toroczkai, and N. Wang, "Modelling disease outbreaks in realistic urban social networks," *Nature*, vol. 429, no. 6988, pp. 180–184, May 2004.
[52] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI gym," 2016, *arXiv:1606.01540*. [Online]. Available: http://arxiv.org/abs/1606.01540
[53] A. Weller, "Bethe and related pairwise entropy approximations," in *Proc. UAI*, 2015, pp. 942–951.
[54] Z. Xu, W. Dong, and S. N. Srihari, "Using social dynamics to make individual predictions: variational inference with a stochastic kinetic model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2783–2791.
[55] L. Fang, F. Yang, W. Dong, T. Guan, and C. Qiao, "Expectation propagation with stochastic kinetic model in complex interaction systems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2029–2039.
[56] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1071–1079.
[57] W. H. Montgomery and S. Levine, "Guided policy search via approximate mirror descent," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4008–4016.
[58] J. Schulman, N. Heess, T. Weber, and P. Abbeel, "Gradient estimation using stochastic computation graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3528–3536.
[59] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: https://arxiv.org/abs/1509.02971
[60] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. 31st Int. Conf. Int. Conf. Mach. Learn.*, vol. 32, 2014, pp. I-387–I-395.
[61] D. Ziemke, K. Nagel, and C. Bhat, "Integrating CEMDAP and MATSIM to increase the transferability of transport demand models," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2493, no. 1, pp. 117–125, Jan. 2015.
[62] B. Kickhöfer, D. Hosse, K. Turner, and A. Tirachini. (2016). *Creating an open Matsim Scenario From Open Data: The Case of Santiago de Chile*. [Online]. Available: http://www.vsp.tuberline.de/publication:TU Berlin,Transp.Syst.PlanningTransp. Telematics
[63] Y.-A. de Montjoye, Z. Smoreda, R. Trinquart, C. Ziemlicki, and V. D. Blondel, "D4D-senegal: The second mobile phone data for development challenge," 2014, *arXiv:1407.4885*. [Online]. Available: http://arxiv.org/abs/1407.4885
[64] C. R. Bhat, J. Y. Guo, S. Srinivasan, and A. Sivakumar, "Comprehensive econometric microsimulator for daily activity-travel patterns," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1894, no. 1, pp. 57–66, Jan. 2004.
[65] G. Flötteröd, M. Bierlaire, and K. Nagel, "ArgmaxBayesian demand calibration for dynamic traffic simulations," *Transp. Sci.*, vol. 45, no. 4, pp. 541–561, 2011.

**LUCKYSON KHAIDEM** received the bachelor's degree in computer science and engineering from the PES Institute of Technology. He is currently pursuing the master's degree in computer science from the State University of New York at Buffalo. His research interests include application of machine learning in financial domain, multi-objective optimization, and game theory in astrophysics (exoplanet habitability estimation).

**MASSIMILIANO LUCA** is currently pursuing the Ph.D. degree with the Mobile and Social Computing Laboratory, Fondazione Bruno Kessler, Italy, and the Faculty of Computer Science, University of Bozen-Bolzano. His main research interests include analysis of human-mobility dynamics and design and development of machine learning methods to tackle mobility challenges.

**FAN YANG** (Member, IEEE) is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, State University of New York at Buffalo, USA. His research interests include reinforcement learning, imitation learning, modeling, generative models, and probabilistic graphical models.

**ANKIT ANAND** received the bachelor's degree in electronics and communication engineering from the Netaji Subhas Institute of Technology. He is currently pursuing the master's degree in computer science with the State University of New York at Buffalo. His research interests include application of machine learning, deep learning, and reinforcement learning.

**BRUNO LEPRI** (Member, IEEE) was born in Ancona, Italy, in September 1980. He received the Ph.D. degree in computer science from the University of Trento, Italy, in 2009. From 2010 to 2013, he was a Joint Postdoctoral Fellow with the MIT Media Laboratory, Boston, MA, USA, and the Fondazione Bruno Kessler (FBK), Trento, Italy, where he is currently leads the Mobile and Social Computing Laboratory (MobS). He is also the Head of the Research with Data-Pop Alliance, the first think-tank on big data and development co-created by Harvard Humanitarian Initiative, the MIT Media Laboratory, and the Overseas Development Institute. His research has received attention from several international press outlets. His research interests include computational social science, urban computing, network science, machine learning, new models for personal data management, and monetization. He received three years Marie Curie Postdoctoral Fellowship in 2010, the James Chen Annual Award for the Best UMUAI Paper, and the Best Paper Award from ACM Ubicomp in 2014.

**WEN DONG** (Member, IEEE) received the Ph.D. degree from the MIT Media Laboratory. He is currently an Assistant Professor of computer science and engineering (Joint Appointment) with the Institute for Sustainable Transportation and Logistics and the State University of New York at Buffalo. His research interests include developing machine learning and signal processing tools to study the dynamics of large-social systems in situ.