RUTGe: Realistic Urban Traffic Generator for Urban Environments Using Deep Reinforcement Learning and SUMO Simulator

Alberto Bazán Guillén[1][a], Pablo A. Barbecho Bautista[2][b] and Mónica Aguilar Igartua[1][c]
[1]*Networking Engineering Department, Universitat Politècnica de Catalunya (UPC), Barcelona, Spain*
[2]*Departamento de Eléctrica, Electrónica y Telecomunicaciones, Universidad de Cuenca (UCuenca), Cuenca, Ecuador*

Keywords: SUMO Traffic Generation, Smart Cities, Realistic Simulator, Deep Reinforcement Learning.

Abstract: We are witnessing a profound shift in societal and political attitudes, driven by the visible consequences of climate change in urban environments. Urban planners, public transport providers, and traffic managers are urgently reimagining cities to promote sustainable mobility and expand green spaces for pedestrians, bicycles, and scooters. To design more sustainable cities, urban planners require realistic simulation tools to optimize mobility, identify location for car chargers, convert streets to pedestrian zones, and evaluate the impact of alternative configurations. However, realistic traffic profiles are essential to produce meaningful simulation results. Addressing this need, we propose a traffic generator based on deep reinforcement learning integrated with the SUMO simulator. This tool learns to generate an instantaneous number of vehicles throughout the day, aligning closely with the target profiles observed at the traffic monitoring stations. Our approach generates accurate 24-hour traffic patterns for any city using minimal statistical data, achieving higher accuracy compared to existing alternatives. In particular, our proposal demonstrates a highly accurate 24-hour traffic adjustment, with the generated traffic deviating only by about 5% from the real target traffic. This performance significantly exceeds that of current SUMO tools like *RouteSampler*, which struggle to accurately follow the total daily traffic curve, especially during peak hours when severe traffic congestion occurs.

1 INTRODUCTION

The emergence of Smart Cities has transformed how urban areas tackle challenges like traffic congestion, sustainable mobility, and the improvement of public transportation services. By leveraging advanced technologies and data analytics, these intelligent cities optimize resources and promote more efficient, sustainable mobility. In this context, Intelligent Transportation Systems (ITS) have become essential for managing traffic, reducing emissions, and enhancing user experiences. However, the effective deployment and testing of ITS demand accurate simulation environments capable of replicating real-world urban traffic patterns.

Mobility hubs, urban planners, traffic engineers, and public transport providers play a crucial role in shaping sustainable and livable cities. The reduction of CO_2 emissions, the enhancement of urban mobility, and the creation of green spaces, such as the innovative superblocks seen in cities like Barcelona, Paris, Bremen, and Bergen, require thorough planning and evaluation. To achieve these goals, realistic simulation tools are essential for testing proposals aimed at improving urban mobility.

In urban simulation research, open source software such as SUMO (Simulation of Urban Mobility) (Lopez et al., 2018) has become a versatile and scalable tool. By offering an open framework, SUMO allows researchers to access its core modeling components, integrate advanced techniques like reinforcement learning, and tailor the system to the specific needs of individual projects. This makes it a key resource for traffic studies requiring the replication of behaviors in complex urban road networks.

Effective urban simulators not only model road infrastructure, but also incorporate dynamic traffic behaviors specific to urban settings. SUMO, as a widely used open-source traffic simulator, provides a flexible platform for modeling urban traffic. However, their default tools for generating a 24-hour traffic intensity pattern might not be sufficiently realistic to evaluate

[a] https://orcid.org/0000-0001-8634-6907
[b] https://orcid.org/0000-0002-5281-9208
[c] https://orcid.org/0000-0002-6518-888X

Bazán Guillén, A., Barbecho Bautista, P. A. and Aguilar Igartua, M.
RUTGe: Realistic Urban Traffic Generator for Urban Environments Using Deep Reinforcement Learning and SUMO Simulator.
DOI: 10.5220/0013375000003941
In *Proceedings of the 11th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2025)*, pages 557-564
ISBN: 978-989-758-745-0; ISSN: 2184-495X

the performance of proposed mobility strategies under varying traffic conditions. Real-world scenarios involve dynamic traffic patterns influenced by weekdays, weekends, holidays, and variations across the 24-hour cycle. A simple traffic profile fails to capture these complexities, limiting the usefulness of simulations for practical applications.

To contribute to improving the generation process of urban traffic, this paper proposes a methodology to automatically generate realistic traffic patterns for any city, using minimal statistical data (e.g., average traffic intensity over a 24-hour period). We have developed and implemented a SUMO-based tool named RUTGe (Realistic Urban Traffic Generator) that leverages Deep Reinforcement Learning (DRL) to learn how to generate the instantaneous number of vehicles throughout the day, aligning closely with the target profiles observed at traffic monitoring stations. Our proposal delivers more accurate and faster results compared to other methods developed for SUMO to generate variable 24-hour traffic profiles.

The rest of the paper is organized as follows. Section 2 outlines the main works related to our study. Section 3 presents the fundamentals of our proposed framework and describes the main algorithms developed. In Section 4, we discuss simulation results. Finally, Section 5 concludes the paper and suggests directions for future research.

2 RELATED WORK

Simulation tools like SUMMIT (Cai et al., 2020) exemplify how urban simulations integrate road infrastructure and dynamic traffic behaviors unique to urban contexts, having been employed to assess autonomous driving systems in densely populated and unregulated traffic environments where vehicle-pedestrian interactions might be highly complex. Similarly, SceneGen (Tan et al., 2021) has demonstrated the effectiveness of auto-regressive neural networks in generating realistic traffic scenarios without relying on predefined rule-based heuristics. This approach is particularly valuable for modeling self-driving vehicles, by addressing traditional challenges in capturing the complexities of urban traffic.

Additionally, the integration of real-world data has proven crucial for advancing urban traffic simulations. For instance, a Markov-chain traffic model was utilized in (Arias et al., 2017) to predict electric vehicle (EV) charging demand at fast-charging stations in urban areas. This model incorporates real-time traffic data obtained from closed-circuit television (CCTV) cameras in Seoul, South Korea, to develop an EV charging-power demand framework. This approach highlights spatial and temporal traffic patterns and the value of context-aware simulations, emphasizing the need for adaptable, data-driven tools tailored to urban environments.

The integration of human preferences and behavior has also been pivotal in advancing traffic simulations. The proposal (Cao et al., 2024) combines reinforcement learning with human feedback to train agents that not only replicate realistic human behavior but also comply with traffic regulations, thereby enhancing the realism of existing traffic models. This approach underscores the significance of incorporating human judgment into the design of traffic simulations to more accurately mirror real-world conditions.

The approach (Padrón et al., 2023) leverages real-world data by developing a traffic model that utilizes real-time data from induction loop detectors installed throughout the city. This model predicts traffic flow and generates realistic Origin-Destination (OD) traffic matrices, achieving more accurate route lengths and a better distribution of traffic sources and destinations compared to the `dfrouter` tool available in SUMO (Lopez et al., 2018).

All these studies collectively highlight the importance of integrating data-driven and human-centered approaches to develop realistic urban traffic simulations. Building on this foundation, our work takes a step further by addressing the challenge of generating a realistic urban traffic profile over a complete 24-hour period corresponding to a typical working day. To achieve this, we have developed a Deep Reinforcement Learning model capable of accurately capturing and replicating the dynamic traffic patterns of a city over the course of an entire day.

3 PROPOSAL TO GENERATE REALISTIC SUMO TRAFFIC

SUMO includes a tool called `dfrouter` (Lopez et al., 2018), designed to generate traffic based on data collected from detection points, such as induction loops. However, this tool somehow fails when applied to urban areas, as it was originally intended for highway scenarios. An alternative approach in (Padrón et al., 2023) improves `dfrouter`'s performance in urban environments, specifically Valencia, Spain, by generating realistic traffic with route lengths suited to the characteristics of the city.

However, to the best of our knowledge, no existing solution aims to generate realistic traffic for a full day for any city solely based on its map and pre-collected traffic data while leveraging reinforcement

learning. Our proposal aims to develop a reinforcement learning-based approach using the SUMO simulator to create a realistic traffic model capable of producing the desired average hourly vehicle flow for a given city. Moreover, our DRL model can be seamlessly extended to cover 24-hour periods, enabling the generation of traffic patterns that not only align with historical averages from traffic metering stations, such as induction loops, but also adapt to specific scenarios, including holidays, weekends, and peak hours.

Our objective is to generate realistic 24-hour traffic patterns representative of an average day in any given city. To achieve this, the process begins with a setup phase that involves preparing the simulation environment. This includes selecting the city map and obtaining real-world traffic measurements for analysis. For our case study, we focused on Barcelona, Spain, using OpenStreetMap (OSM) (OpenStreetMap, 2024) data. The map was converted into a SUMO-compatible format using SUMO's `netconvert` tool. Historical traffic intensity data, measured in number of vehicles per hour at various traffic monitoring stations across the city, was provided by the Barcelona City Council. From this dataset, we selected data from five key monitoring stations. These locations were replicated in our SUMO simulation scenario by placing `induction loops` at the same positions as the real-world sensors that collected the historical data.

After the configuration phase, the next step involved developing a realistic traffic generator tailored to the specific scenario. This traffic-generating agent is based on a reinforcement learning model designed to create traffic patterns that closely align with a predefined target traffic profile for a given hour. Following a necessary training period, the model can accurately generate the desired traffic for the specified hour, establishing the agent as a key component of the realistic traffic simulation system for the city.

Once the traffic generation model is trained, it will be executed in SUMO for each of the 24 hours to simulate a complete daily traffic profile. The model generates the required traffic for each hour while considering residual traffic from preceding hours, which may influence subsequent traffic conditions. Upon completing the 24-hour simulation cycle, the system exports a file containing vehicle data and routes that replicate the desired traffic intensity profile for the city on a typical working day, as it represents the most representative and congested traffic conditions.

3.1 Deep Reinforcement Learning for Realistic Traffic Simulation

We have developed a model-free Deep Reinforcement Learning approach to generate realistic urban traffic patterns. SUMO is used as the simulation environment, acting as a black-box traffic model that provides state observations as output after simulating one hour of traffic. The RL agent's objective is to learn a policy that dynamically adjusts the number of vehicles introduced into the network during the simulation, minimizing the deviation between the generated traffic and the target traffic. This section outlines the formalization of the RL framework, its components, and the implementation methodology.

3.1.1 Model-Free Reinforcement Learning Framework, 1-Hour Agent

Reinforcement learning is a paradigm in which an agent interacts with its environment to learn behavior that maximizes a cumulative reward. In the model-free approach, the agent does not attempt to construct or predict the transition dynamics of the environment. Instead, it learns a policy directly based on observed states, actions, and rewards.

The proposed RL system is framed as a Markov Decision Process (MDP), defined by the tuple (S,A,R,P), where S is the set of states s observed from the environment, A is the set of actions a the agent can take, $R(s,a)$ is the reward function providing feedback for taking action a in state s, and $P(s'|s,a)$ is the transition probability function (from state s to state s'), which is implicit in the model-free RL and learned indirectly.

In our implementation, SUMO served as the environment, and its outputs after one hour of simulation provided the observations necessary to define the state and calculate the reward. The RL framework was instantiated with the following four components:

a. Environment: SUMO Simulator
SUMO is widely used as a high-fidelity simulation environment for urban traffic. In our simulation scenario, after each simulation step (representing one execution during the simulated hour), SUMO generates traffic metrics such as vehicle intensity (vehicles/h), congestion level (vehicles/km^2), and average speed (km/h), among others. In our case, vehicle intensity is used to compute the current observation of the environment and subsequently calculate the reward $R(w,a)$ for the selected action.

b. State, s
The state in the reinforcement learning framework encapsulates the traffic injection configuration for the

simulation. Specifically, the state consists of the number of vehicles to be introduced into the traffic network, distributed throughout the hour being analyzed in the SUMO simulation environment.

The injection of traffic, specifically the Origin-Destination (OD) trips, plays a crucial role in defining the traffic distribution within the simulation environment. The number of vehicles to be introduced into the network determines the number of OD pairs that will be distributed across the map. These OD pairs are used to generate trips via the *OD2Trips* tool in SUMO, which converts OD pairs into individual trips within the network.

To ensure a realistic traffic flow that aligns with the real-world distribution of vehicles, trips are distributed across the network using the `via` attribute, which allows specifying the exact routes that vehicles will follow. Subsequently, the `duarouter` tool in SUMO is used to generate the routes for each trip. These routes represent the paths that vehicles will take during the simulation. The selected tools were inspired by the work (Barbecho Bautista et al., 2022).

Once the trips and their respective routes are generated, they form the basis for running the simulation in SUMO. The simulation outputs consist of intensity measurements recorded by induction loops placed on the map throughout the simulated hour. These outputs serve as the observed states in the reinforcement learning framework and play a crucial role in evaluating the agent's performance. Additionally, they guide policy updates during training, ensuring that the simulated traffic conditions accurately reflect real-world traffic patterns.

To improve the efficiency and stability of reinforcement learning, the state s was normalized to $[0,1]$ using the minimum and maximum values observed for each traffic metric. This ensured consistent scaling and prevented any feature from dominating the learning process.

c. Action, a

Actions are defined as the number of vehicles injected into the traffic network at the start of each simulation step. The agent learns to determine the optimal injection rate to achieve the desired traffic conditions. The action A in the proposed reinforcement learning framework modifies the current state S to produce a new state S'. This modification is defined as:

$$S' = S \cdot A, \tag{1}$$

where A is a continuous scalar value within the range $[-0.5, 2.0]$. This design allows the agent to dynamically scale the current state, enabling reductions ($A < 1$) and amplifications ($A > 1$) of the traffic metrics.

After applying the action, the new state S' is evaluated to compute the reward. This reward, along with S', the action A, and the original state S, is stored in the experience buffer for training the Proximal Policy Optimization (PPO) algorithm.

By iteratively adjusting the state through actions, the agent learns to select optimal scaling factors (A) that minimize the deviation from the desired traffic conditions.

d. Reward, $R(s,a)$

The reward is calculated as the negative mean squared error (MSE) between the target traffic metrics and the corresponding traffic output generated by SUMO. The use of a negative value ensures that the reinforcement learning agent interprets smaller deviations as higher rewards, aligning with the objective of minimizing the error. The reward function is formally defined as:

$$R(s,a) = -\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{2}$$

where y_i represents the target value for a traffic metric and $\hat{y}_i$ is the observed SUMO value for a given state s and action a. This reward helps the agent minimize the deviation between the current traffic conditions and the desired traffic pattern.

The SUMO simulator outputs ($\hat{y}_i$) the average traffic intensity values recorded by the virtual detectors placed within the simulated map. These detectors are placed at specific locations that correspond to real-world traffic detectors, allowing direct comparisons between actual and simulated traffic conditions.

The target values (y_i) represent the average hourly traffic intensities recorded during a typical weekday, provided by the Barcelona City Council. These targets are based on real-world data collected from detectors located at the same positions as the virtual ones in the simulation.

To ensure adaptability, the reinforcement learning model is trained using a range of target values representing various traffic scenarios. This approach enables the model to generalize and adjust the simulated traffic for any future demand. By minimizing the deviation between the simulated and target intensities, the model learns to dynamically regulate traffic injection rates to match desired patterns within a one-hour simulation period.

The target intensities are integrated into the reward function as described in Eq. (2), where the negative mean squared error (MSE) between the SUMO outputs and the targets guides the agent to improve simulation accuracy and realism.

Additional Reward Components. To enhance the learning process, the reward function incorporates the

following elements:

- **Step Penalty:** At every step, the agent receives a small negative reward, $R_{\text{step}} = -\lambda$, where $\lambda > 0$. This penalty encourages the agent to find solutions in fewer steps, promoting efficiency.
- **Goal Achievement Reward:** If the agent achieves an MSE smaller than a predefined threshold (MSE $< 10^{-5}$), it is granted a large positive reward, $R_{\text{goal}} = +\eta$ ($\eta \gg 1$). This reward strongly reinforces successful policies that achieve the objective with high precision.

Complete Reward Function. The complete reward at each step, $R_{\text{total}}(s,a)$, integrates these components and is defined as follows:

$$R_{\text{total}}(s,a) = \begin{cases} R(s,a) - \lambda, & \text{if MSE} \geq 10^{-5} \\ +\eta, & \text{if MSE} < 10^{-5} \end{cases} \quad (3)$$

To balance the incentive for achieving the objective with the penalty for prolonged episodes, the step penalty and the goal achievement reward were defined as follows:

- **Step Penalty (λ):** A value of $\lambda = 0.01$ was used to lightly penalize each step, encouraging the agent to seek efficient solutions without overshadowing the primary objective.
- **Goal Achievement Reward (η):** A large positive reward of $\eta = 10.0$ was granted when the agent achieved a mean squared error smaller than 0.00001. This strong positive signal prioritized policies that met the desired precision.

These values ensured a balance between promoting efficient behavior and strongly reinforcing successful outcomes.

3.1.2 Policy, Step Function, and Training Process

The agent's policy is trained using a model-free reinforcement learning framework based on the Proximal Policy Optimization (PPO) algorithm. PPO is well-suited for tasks involving complex environments such as SUMO due to its stability and efficiency. The agent iteratively updates its policy to maximize cumulative rewards through controlled updates, ensuring effective exploration and exploitation.

A custom step function integrates SUMO with the RL framework, facilitating interaction between the agent and the simulation environment. For each step:

- The agent takes an action a, modifying the vehicle injection rate in SUMO.
- SUMO simulates one hour of traffic and provides updated metrics that define the next state s'.
- The reward $R(s,a)$ is calculated using the mean squared error (MSE) between the target and simulated traffic metrics and also taking the additional reward components, as defined in Eq. (3).

The training process consists of multiple episodes, each comprising several one-hour simulations, each time with different target values. During each episode:

- The agent observes the current state s and calls SUMO to obtain the observation from the SUMO outputs.
- The step function calculates the reward and transitions the system to the next state s'.
- The PPO algorithm updates the policy based on cumulative rewards, enabling the agent to minimize the deviation from the target traffic metrics.

This iterative process ensures that the agent learns a robust policy for dynamically adjusting traffic conditions to achieve realistic and efficient simulations. Additionally, using different target values in each simulation helps develop an agent capable of adapting to a wide range of targets.

3.2 24-Hours Traffic Generator

Once the 1-hour traffic model is trained, it is utilized to generate the desired traffic pattern over a 24-hour period. For each hour, the 1-hour DRL-based model described in Sec. 3.1 is executed with the target traffic intensity for that specific hour, as outlined in Algorithm 2. At the end of the simulation, the residual traffic is collected. This residual traffic represents vehicles remaining on the network after injecting the desired number of vehicles at the start of the given hour. These delayed vehicles contribute to the traffic intensity of subsequent hours, serving as additional input for the following simulations.

The residual traffic is accumulated and subtracted from the target value for the subsequent hour, under the assumption that the carried-over traffic from the previous hour will already be present. This approach fixing the traffic for one hour, accounting for residual traffic from the previous hour, and subtracting it from the target for the next hour is applied iteratively throughout the 24-hour period, as detailed in Algorithm 1. This process ensures the generation of a traffic profile that closely approximates the desired pattern for the entire day.

As a result, a set of vehicles and routes is generated with varying departure times and locations,

Algorithm 1: 24h-TGA ⇒ 24-hours Traffic Generation Algorithm.

Data: Map file (`.net.xml`), start time T_{start}, end time T_{end}, hourly target intensities `target[]`
Result: Complete route file (`final_routes.rou.xml`)

Initialize `final_routes.rou.xml` as an empty file;
for $i \leftarrow T_{start}$ **to** T_{end} **do**
  `// Generate traffic for hour` i
  `[routes.rou.xml, residual[]] = 1h-TGA(`i`, target[`i`])` ← Alg. 2;
  Append `routes.rou.xml` to `final_routes.rou.xml`;
  `// Adjust targets for subsequent hours`
  for $j \leftarrow i+1$ **to** T_{end} **do**
    `target[`j`] = target[`j`] - residual[`j`];`
return `final_routes.rou.xml`;

Algorithm 2: 1h-TGA ⇒ 1-hour Traffic Generation Algorithm.

Data: Hour i, target intensity `target[`i`]`
Result: Route file (`routes.rou.xml`), traffic residual vector (`residual[]`)

Initialize `routes.rou.xml` as an empty file;
Initialize `residual[]` as a vector of zeros;
while *MSE(observed intensity,* `target[`i`]`*)* $>$ *threshold* **do**
  Call PPO model with `target[`i`]`, state S to propose action a and adjust traffic injection;
  Update observed traffic intensity and calculate $R(s,a)$;
  Make $S' = S$;
Compute `residual[]` as the traffic delayed to subsequent hours;
return `routes.rou.xml`, `residual[]`;

populating the map with vehicles that undertake trips throughout the simulation. This method achieves the desired average hourly traffic intensity, enabling the creation of realistic, time-dependent traffic flows that reflect real-world patterns and maintain consistency over the 24-hour simulation period.

4 PERFORMANCE EVALUATION

The proposed tool demonstrates remarkable versatility, allowing the generation of realistic traffic patterns for any city map available in OpenStreetMaps (OpenStreetMap, 2024). By integrating traffic intensity data for specific detection points in the city, the tool produces simulated traffic that matches the average traffic intensity observed at these points. This capability enables users to replicate realistic traffic conditions according to their needs, making the tool highly adaptable for diverse simulation scenarios. To demonstrate our methodology, we used the Barcelona simulation scenario provided by OpenStreetMap (OSM) (OpenStreetMap, 2024), depicted in Fig. 1. In our study, we selected five out of the fifteen available traffic stations in Barcelona to demonstrate the methodology and evaluate the accuracy obtained.

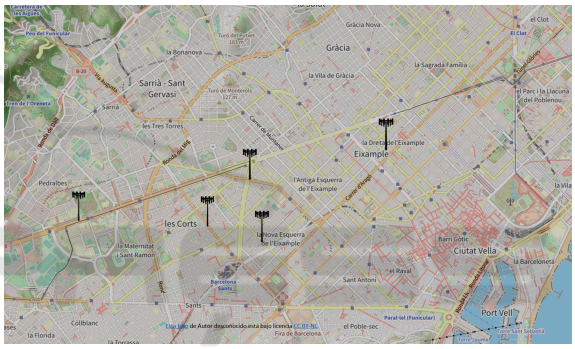


Figure 1: Simulation scenario of a 45 km^2 area in Barcelona (9 km x 5 km), with traffic intensity monitored at five key traffic stations (indicated by black icons). Map sourced from OSM (OpenStreetMap, 2024).

The RUTGe tool leverages our pre-trained DRL-based model, which requires an initial investment of near 5 hours in training time, with a computer with these features: CPU i9-12900K, 3.2 Ghz, 30MB; RAM 32GB DDR5 4800Mhz. Once trained, our RUTGe tool efficiently generates traffic for the specified time range, averaging just 15 minutes per simulated 24 hours of real-world traffic. Users can customize the time range and adjust traffic intensity targets for each hour to meet specific traffic profiles. This underscores the tool's efficiency, showing that the initial investment in training time is well worth the valuable outcomes it delivers. Note that current SUMO tools, such as *RouteSampler*, require around 1 hour 15 min to simulate 24 hours of traffic.

4.1 Simulation Results

The progression of the reward over iterations for a single hour is shown in Fig. 2. The selected hour (9:00

AM) was the peak traffic hour during the day. We observe that the reward achieved during each simulation step consistently improves as the model iteratively refines its policy to match the desired traffic intensity. The graph shows a steady increase in reward until it converges to the maximum value, indicating that the simulated traffic matches the target intensity.

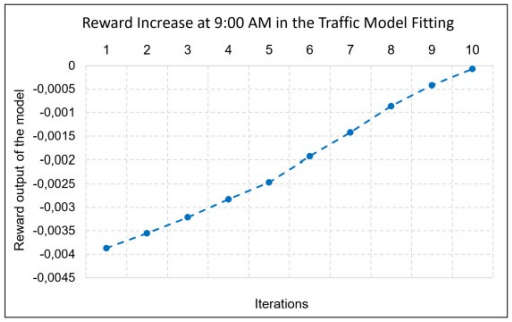


Figure 2: Reward progression during traffic generation for a single hour (9:00 AM).

To validate the accuracy of the tool, simulations were performed for a 24-hour urban scenario. Confidence intervals of 99%, computed from 10 repetitions with different seeds, are shown. These intervals are very small, indicating consistency in the generated models. Figs. 3a and 3b compare the simulated traffic intensities (dashed red lines) with real-world data (solid blue lines) for each hour. Results indicate that the traffic generated by our RUTGe tool aligns more closely with the target traffic intensity compared to RouteSampler, highlighting the superior precision of our traffic generation tool. The average relative error is 5.5% with RUTGe and 8.5% with RouteSampler. The main difference is that our approach takes into account the residual traffic from each hour, which impacts subsequent hours (see Sec. 3.2), whereas RouteSampler does not.

Figure 4a presents the spatial traffic distribution visualized on the colormap generated using the SUMO tool `plot_net_dump.py` for a 24-hour simulation. The generated traffic aligns well with real traffic patterns on the most congested main roads, while also being evenly distributed across the remaining road networks, effectively reflecting realistic traffic behavior. This result underscores the ability of the RUTGe tool to model complex urban traffic systems with high fidelity. Fig. 4b shows that current SUMO tools, such as *RouteSampler*, tend to concentrate the majority of traffic around traffic stations to meet the target traffic intensity. However, the traffic distribution across the rest of the scenario is less evenly spread.

5 CONCLUSIONS AND FUTURE WORK

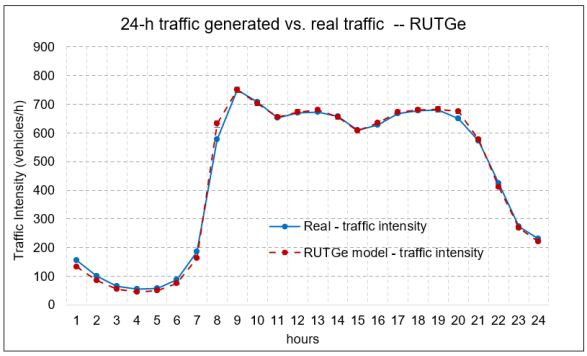
This work presents the development of a tool for generating realistic urban traffic patterns using deep reinforcement learning. By utilizing a city map extracted from OpenStreetMap and historical traffic intensity data, the tool trains a reinforcement learning agent capable of achieving the desired average traffic intensity for a one-hour simulation. Once trained, the model can extend its functionality to simulate multiple hours (e.g., a day) with varying traffic intensity targets, aiming to generate traffic patterns that closely resemble real-world conditions.

For validation, the tool was applied to a 45 km^2 section of Barcelona to simulate 24 hours of traffic on an average working day. The results demonstrate promising accuracy, achieving a relative error of 5.5% compared to real-world traffic intensity data. Additionally, the short average execution time of 15.2 minutes per simulation justifies the initial training time of 5 hours, as the trained model can be reused to simulate any combination of hours and intensity targets within the same environment. Furthermore, the traffic distribution achieved on the map aligns well with realistic urban patterns, making the tool valuable for simulating various services and incidents in urban environments. The generated traffic patterns can serve as the foundation for advanced studies in intelligent transportation systems, urban planning, and autonomous vehicle testing.

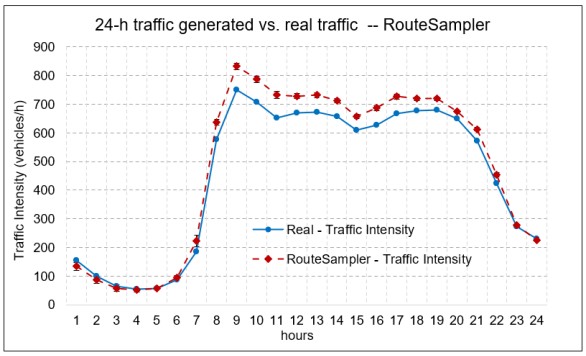
As future work, we plan to develop a Federated Deep Reinforcement Learning scheme, where each independent traffic station will locally train its own model. Stations will share hyperparameters, collaboratively constructing a global prediction model. This approach is expected to significantly reduce training time while maintaining model accuracy.

ACKNOWLEDGEMENTS

This work was partially supported by the Spanish Government under these research projects funded by MCIN/AEI/10.13039/501100011033: DISCOVERY PID2023-148716OB-C32; MOBILYTICS TED2021-129782B-I00 (also funded by the European Union NextGenerationEU/PRTR); COMPROMISE PID2020-113795RB-C31; predoctoral scholarship associated with the ”Generación de Conocimiento” Projects, Call 2022, PRE2021-099830. Also, by the Generalitat de Catalunya AGAUR grant ”2021 SGR 01413”.

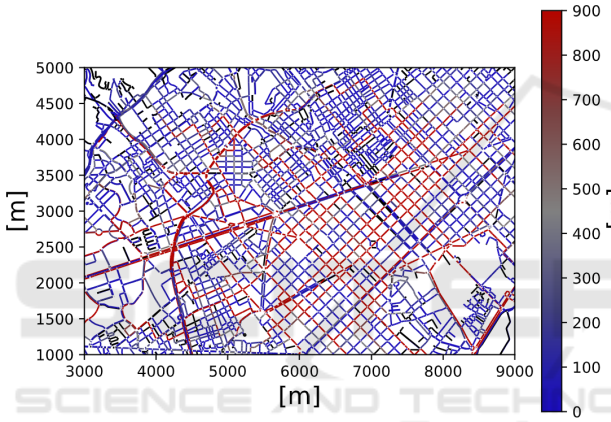


(a) 24-hours traffic generated using our proposal RUTGe.

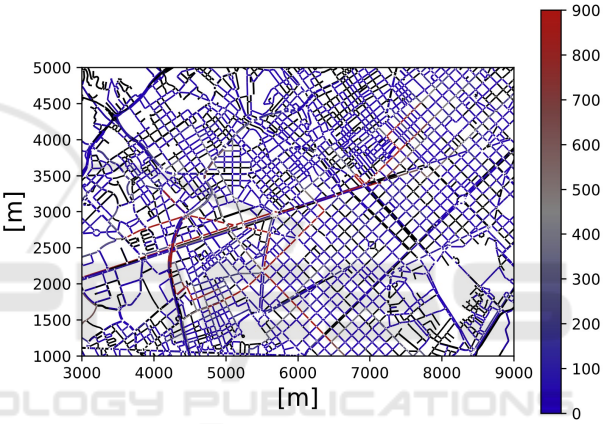


(b) 24-hours traffic generated using the *RouteSampler* tool.

Figure 3: Comparison of simulated traffic intensities with real-world data (target traffic to adjust the model). 99% confidence intervals are shown in the RUTGe model and RouteSampler tool. The average relative error is 5.5% and 8.5%, respectively. The simulation time is 15 min and 19 min, respectively.



(a) SUMO colormap using our proposal RUTGe.



(b) SUMO colormap using the *RouteSampler* tool.

Figure 4: Traffic distribution across the map for a 24-hour simulation. Urban scenario depicted in Fig. 1.

REFERENCES

Arias, M. B., Kim, M., and Bae, S. (2017). Prediction of electric vehicle charging-power demand in realistic urban traffic networks. *Applied energy*, 195:738–753.

Barbecho Bautista, P., Urquiza Aguiar, L., and Aguilar Igartua, M. (2022). How does the traffic behavior change by using sumo traffic generation tools. *Computer Communications*, 181:1–13.

Cai, P., Lee, Y., Luo, Y., and Hsu, D. (2020). SUMMIT: A simulator for urban driving in massive mixed traffic. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4023–4029. IEEE.

Cao, Y., Ivanovic, B., Xiao, C., and Pavone, M. (2024). Reinforcement learning with human feedback for realistic traffic simulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14428–14434. IEEE.

Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wiessner, E. (2018). Microscopic Traffic Simulation using SUMO. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2575–2582.

OpenStreetMap (2024). https://www.openstreetmap.org.

Padrón, J. D., Hernández-Orallo, E., Calafate, C. T., Soler, D., Cano, J.-C., and Manzoni, P. (2023). Realistic traffic model for urban environments based on induction loop data. *Simulation Modelling Practice and Theory*, 125:102742.

Tan, S., Wong, K., Wang, S., Manivasagam, S., Ren, M., and Urtasun, R. (2021). SceneGen: Learning to Generate Realistic Traffic Scenes . In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 892–901.