Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 184 (2021) 745–752

Procedia Computer Science

www.elsevier.com/locate/procedia

The 10th International Workshop on Agent-based Mobility, Traffic and Transportation Models, Methodologies and Applications (ABMTRANS 2021)
March 23 - 26, 2021, Warsaw, Poland

Automated generation of traffic signals and lanes for MATSim based on OpenStreetMap

Theresa Ziemke[a,*], Söhnke Braun[a]

[a]*Technische Universität Berlin, Transport Systems Planning and Transport Telematics, Salzufer 17-19, 10587 Berlin, Germany*

Abstract

OpenStreetMap (OSM) offers information on traffic networks like the position of intersections and streets, road categories, speed limits, number of lanes etc. Based on this information, network input data for transport simulations, e.g. the agent-based transport simulation MATSim can be generated. A MATSim network is modeled as a graph with nodes (intersections) and links (streets). Previously, data about traffic signals and lanes could only be provided to MATSim simulations by manual data preparation or had to be emulated by reducing flow capacities of links that lead towards signalized intersections. To be able to model inner-city traffic more realistically, this study extends the input generation of MATSim. Information about traffic signals and lanes is extracted from OSM and automatically processed to create signal and lane input files for the simulation. Processing data about traffic signals and lanes is required to optimize fixed-time plans, as well as to develop and validate algorithms for traffic-adaptive signals. This study provides the basis for further development of traffic lights in the transport simulation MATSim. An application of a MATSim simulation including traffic signals and lanes in the Berlin metropolitan area verifies the developed method.

Keywords: OpenStreetMap; agent-based transport model; traffic signals; lanes; scenario creation

1. Introduction

Traffic models aim to represent the actual traffic state and give predictions on the effect of planning measures. Traffic and city planning without such models is not imaginable nowadays. The multi-agent transport simulation MATSim [5] considered here is particularly useful to model aspects that are highly influenced by individual decisions. It runs large-scale real-world scenarios in reasonable time. Of course, input data representing the model region is necessary to built a scenario. Part of this is the information about the street network. In MATSim, this is represented by a directed graph with nodes (intersections) and links (streets). OSM has proven to be a valuable data source for

* Corresponding author. Tel.: +49-30-31478783 ; fax: +49-30-31426269.
E-mail address: tziemke@vsp.tu-berlin.de

1877-0509
10.1016/j.procs.2021.03.093

road networks [13]. It offers information about traffic networks as the position of intersections and streets, street categories, speed limits, number of lanes etc. Based on this information, the openly available *OsmNetworkReader* automatically creates a MATSim network file for any specified region. However, it does not contain any information regarding traffic signals or lanes. Instead, the number of lanes is only used to estimate link capacities and the missing intersection modeling is emulated by reducing link flow capacities. This is because the explicit simulation of traffic signals and lanes is optional in MATSim. To be able to explicitly model traffic signals and lanes, further input files are necessary in MATSim. These are quite data-hungry – they need information about the lanes on all links, their storage and flow capacity, and to which next link they lead to. Additionally, the position of signals on links or lanes is needed, as well as there signal control scheme and the information which lanes are signalized jointly. Thus, a manual creation of this input is cumbersome. An automatic generation tool was missing so far. Possibly, this is the reason why only a few MATSim scenarios include specific modeling of traffic signals and lanes, so far.

Therefore, this study provides an automated generation tool for traffic signals and lanes input data to model arbitrary study areas in MATSim based on OSM data. This tool extends the standard *OsmNetworkReader*. As both – lane and signal input – are separate files, and the MATSim lane input file is independent from the information on traffic signals, one can also use this tool to only model lanes without traffic signals. The present paper is organized as follows: First, Sec. 2 gives some background information on OSM, MATSim, and the standard *OsmNetworkReader*. Sec. 3 describes the new *OsmNetworkSignalAndLanesReader*. An application of the functionality is given in Sec. 4 by generating signals and lanes data for the Open Berlin Scenario. This verifies the usability of the automatic generation tool and reveals the challenges one has to tackle when using it (see Sec. 5). Sec. 6 concludes and describes possible next steps.

2. Background

2.1. OpenStreetMap (OSM) data

OpenStreetMap (OSM) provides free geographic data for the world. It is editable and being built by volunteers. With this, its accuracy depends on the interest of people to provide information about their neighborhood. Starting with mapping the UK in 2004, high-quality maps of large parts of the world are now provided and used.

OSM provides data about buildings, facilities, land use, trees, letter boxes, public transport lines, and many more. For this study, only data regarding the road network is used: Ways (that represent streets), nodes (representing intersections) and relations (containing information regarding two ways, e.g. turn restrictions from one way to the other). Of particular interest for the generation of lane and traffic signal information in this study are the following tags:

- the number of lanes (*lanes*, *lanes:forward/backward*),
- information on lane directions (*turn:lanes:forward/backward*,
- link-specific signal positions (*highway=traffic_signals, highway=crossing*), and
- information about turn restrictions (*relation:restriction*).

Note that the length of a lane is not directly stated by OSM. Also, no information is given which lanes are signalized separately. An example for the information contained in OSM can be found in Fig. 1.

2.2. Multi-agent transport simulation MATSim

The multi-agent transport simulation MATSim [5] is an open-source simulation tool that is widely used to model real-world traffic scenarios in reasonable time. In MATSim, travelers are modeled as agents that follow a daily plan of activities. While traveling between activities they can be delayed by other travelers. In an iterative setting, agents may modify their plans to improve their utility (see Fig. 2). MATSim's co-evolutionary algorithm aims a situation where no agent can improve by unilaterally changing their plan (stochastic user equilibrium). While being fully microscopic on the agent-side, MATSim's flow model is simplified to guarantee reasonable runtimes also for large scenarios.

MATSim's mobility simulation runs on a directed graph that represents the road network with nodes (i.e. intersections) and links (i.e. streets). Traffic dynamics are modeled on links only; no intersection logic is modeled in the standard framework. Links operate as simple first-in-first-out (FIFO) waiting queues without car following and lane

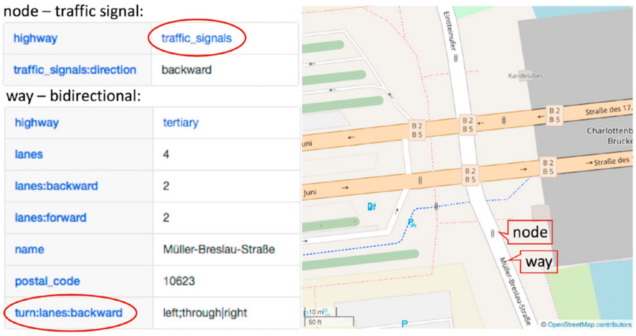

Fig. 1. Example of traffic signals and lanes data in OSM.

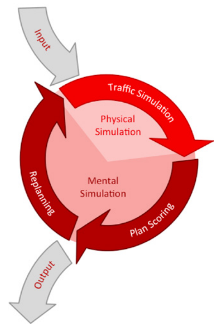

Fig. 2. Iterative co-evolutionary algorithm of MATSim. Source: [11]

Fig. 3. Separate FIFO-queues for lanes in MATSim (left side) and turn specific spill back at traffic signals in MATSim (right side). Source: [4]

changing. Still, lanes can be modeled as link-internal, parallel FIFO-queues (as shown in Fig. 3, left). Each lane may have different turn restrictions, flow capacities and lane length. As traffic signals are lane-specific, they enable individual waiting times for different turn directions and realistic spill-back patterns at intersections (see Fig. 3, right). Lanes and traffic signals are modeled as an extension in MATSim and described in more detail by Grether and Thunig [3].

To be able to model lanes and traffic signals, MATSim needs information on the number of lanes per link, link and lane length, lane directions (in form of ID numbers of next links), lane-specific traffic signal positions and information on conflicting directions, i.e. information which signals can show green at the same time. Also, their signal control scheme is needed – either as (adaptive) algorithm or as fixed time schedule.

2.3. Standard generation of MATSim networks

The standard *OsmNetworkReader* was developed by Zilske et. al [13]. It takes OSM data for an arbitrary, given region and converts it into a MATSim network file. The user can specify which hierarchy layers of ways should be included for conversion (to be able to 'thin out' the network and e.g. only simulate bigger streets). Additionally, a set of special node IDs can be provided, that should be definitely kept during the conversion process (e.g. for counting stations). The conversion from OSM to MATSim is done in three steps. Algo. 1 gives an overview on the converting scheme in pseudo code. Firstly, OSM data is prepared: All nodes and ways of the study region are read, incomplete ways at the boundary are removed and nodes are filled with additional information like the number of in- and outgoing ways etc. Secondly, the so constructed data is simplified: Only nodes that belong to the given set of special node IDs and nodes that have at least one adjacent way matching the specified hierarchy layers are kept. All other nodes and their adjacent ways are removed. Finally, MATSim nodes and links for all remaining OSM nodes and ways are created. The information on the number of lanes on a link is used to define the flow capacity of a link (i.e. the number of vehicles that are allowed to leave a link per time step). Other than that, no information about lanes nor traffic signals is used by the standard *OsmNetworkReader*.

3. Automated generation of MATSim networks including lanes and traffic signals

To be able to generate MATSim scenarios including lanes and traffic signals automatically, the standard *OsmNetworkReader* (described in Sec. 2.3) has been extended to also create lanes and signal input files. Algo. 2 gives an overview about this extension. The three main categories *prepare OSM data*, *simplify OSM data* and *create MATSim data* are kept. The additional steps noted in Algo. 2 are the main steps within the automated generation of the lanes and

Algorithm 1 Standard OSM to MATSim Conversion

Input: OSM data, hierarchy layers, optional: special node IDs
1: **Prepare OSM data**
2: **Simplify OSM data**
3: **Create MATSim data**
Output: MATSim network

Algorithm 2 Extended OSM to MATSim Conversion with Lanes and Signals

Input: OSM data, hierarchy layers, optional: special node IDs, sub-area bounding-box for which signal data should be processed
1: **Prepare OSM data**
2: Identify signalized intersections
3: Push signals to junctions
4: **Simplify OSM data**
5: Merge complex intersections
6: **Create MATSim data**
7: Assign turn possibilities
8: Provide basis fixed-time plans
Output: MATSim network, lane and signal files

signal information for MATSim. They extend the standard implementation at the end of each category, respectively, and are described in the following subsections.

Most, but not all information that is needed to model lanes and signals in MATSim can be extracted from OSM. The number of lanes per link can be directly taken from OSM. Link and lane lengths can be calculated based on coordinates of adjacent nodes. Lane flow capacities are set like it is done for links in the standard *OsmNetworkReader*. Turn directions of lanes are more complex as they are only given textually in OSM (e.g. *straight* or *slight left*). To be able to translate this into MATSim data, one needs to identify corresponding from-link/to-link pairs. In terms of traffic signals, even more information is missing as OSM only provides link-specific traffic signal positions, but no information regarding which lanes are (or can be) signalized separately. Also, information about conflicting directions (which signals can show green at the same time) has to be deduced based on the general intersection layout solely.

3.1. Identify signalized intersections

Unfortunately, the position of traffic signal tags in OSM is not standardized or, at least, not used uniformly. Fig. 4 gives some examples of how traffic signal are tagged in OSM: In the left case, traffic signals are tagged upstream of the intersections (realistic representation). The right picture shows a case where a logical representation is used and the traffic signals are tagged directly at the intersection point of the ways. Usually, a whole intersection is tagged either with the first or the second style. But also mixed representations exist (see middle picture in Fig. 4).

To be able to automatically translate signal positions from OSM to MATSim, signalized intersections need to be identified as a first step. To do so, we push traffic signals downstream to achieve the logical representation for all intersections. In some cases, the traffic signals thereby even need to be moved into another way (when traffic signal and intersection are located on different ways in OSM). To avoid the generation of errors, a maximal push distance (*signal push threshold*; by default 30m) is specified in the reader.

3.2. Merge complex intersections

In OSM, large roads are usually represented by two parallel ways with opposing directions, leading to complex intersections where two or four nodes represent a single intersection. In MATSim, such a representation does not provide any benefit, as traffic flow is only modelled on links. On the other hand, identifying turn relations, oncoming directions (for realistic grouping of signal green times) etc. is more elaborate at such complex multi-node intersections. Thus, we merge all multi-node intersections into a single node (as shown in Fig. 5) corresponding to the centroid of the previous intersection nodes. To be able to identify complex intersections with more than two nodes (upper case

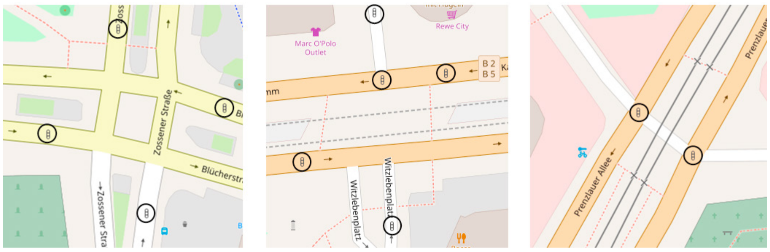

Fig. 4. Different positions of traffic signal tags at signalized intersections in OSM. From left to right: realistic, mixed and logical representation.

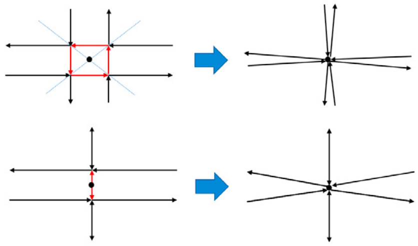

Fig. 5. Complex intersections are simplified to be represented by a single node. Source: [8]

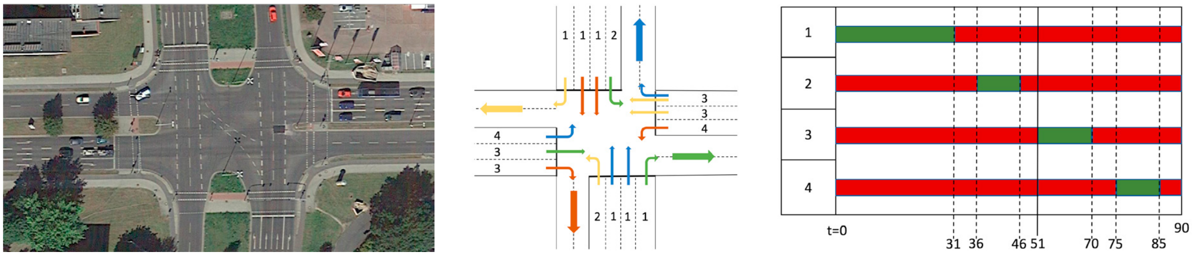

Fig. 6. Example of the assignment of turn possibilities given by OSM (left) to lanes in MATSim (middle; colors illustrate the assignment), and to signal groups (illustrated by the numbers in the middle picture) and corresponding fixed-time base plans (right). Source: [8]

in Fig. 5), the algorithm searches for cycles of uni-directional ways, where each way is shorter than a threshold (*intersection merge threshold*; by default 30m). Two-node intersections (lower part of Fig. 5) are characterized as two nodes on two different uni-directional ways, connected by a bi-directional way shorter than the same threshold.

3.3. Assign turn possibilities

Turn possibilities in OSM are given textually on the lanes (e.g. *straight*, *slight left* etc.). They correspond to the arrows that are usually marked on the surface of the road. Additionally, turn restrictions (from road sings) are given by the *relation:restriction* objects in OSM (e.g. *no left turn*, *no U-turn* etc.). In MATSim, each lane needs the information which links it leads to, i.e. link IDs of possible turns need to be given. To be able to translate the OSM information into the MATSim format, angles to all outgoing links are calculated and evaluated to identify correct to-links of lanes. All possible turns of a link – in form of their textual representation given by OSM – are ordered from right to left to be able to match them with the corresponding outgoing links (similarly ordered based on their angles). As the simplification of multi-node intersections (see Sec. 3.2) would impair this process, all angles are calculated before this step. The left and middle picture of Fig. 6 illustrate the assignment of turn possibilities. Subsequently, the *relation:restriction* information of OSM is evaluated, corresponding link IDs are matched similar as described above. Links that correspond to restricted turns are removed as possible next links of the lanes.

3.4. Provide basis fixed-time plans

Signal data in MATSim consists of three different files. *Signal systems* define the signalized intersections and which signals on which links and lanes belong to it. *Signal groups* collect signals that always show green together, e.g. on straight and right turn lanes. The *signal control* file specifies which signal control algorithm is used and contains fixed-time plans if necessary. To fill this data, the converter creates signals on each lane of all signalized intersections. Due to preparation and simplification, this step is straightforward. The creation of signal groups and signal control is more complex. OSM does not provide any information on how the traffic signals are controlled or which lanes are signalized separately (e.g. whether there exists a separate left turning signal). To still be able to provide an applicable MATSim scenario, we construct default fixed-time plans. Lanes are grouped into signal groups based on the *turn:lanes* information and intersection layout: If a left turning lane exists, a separate signal group for left turns is created which contains the two oncoming left turns. If not, all lanes of the direction and its oncoming direction are put into a joint signal group. The oncoming direction is, thereby, identified based on the angles of the links.

A default signal plan is defined based on German guidelines [1]: As minimum green time, 10 s are used; for 5 s all signals show red between the signal phases; the cycle time for all signal systems is set to 90 s. Additional green time besides the minimum green time is assigned based to the number of lanes as an indicator for high traffic rates. An example of a so constructed fixed-time plan is given in Fig. 6. This signal plans do not necessarily model a realistic signal control. Still, they provide a basis for a simulation with traffic signals that can be easily adapted. For example, one can use the given signal positions and signal groups to run adaptive signal control algorithms. MATSim provides different implementations of adaptive signal control approaches, all with very good performance (see [2, 9, 12]). Generating a scenario based on the presented approach and running it with one or some of these signal control approaches possibly gives a quite realistic model of arbitrary inner-city regions.

The standard *OsmNetworkReader* compensates missing intersection dynamics by reducing flow capacities on links (which are set based on the hierarchy level and the number of lanes). This is no longer necessary when signals are explicitly modeled. The extended reader, therefore, uses the saturation flow rates as flow capacities at signalized lanes (which are defined as 2000 veh/h in [1]).

4. Application – Modelling Berlin with Lanes and Traffic Signals

To test and verify the implementation, it was applied to the Berlin metropolitan area. For this purpose, the MATSim Open Berlin Scenario [10] is extended by signal and lane data. Fig. 7 shows some visualizations of the scenario. As the simulation of traffic signals and lanes in MATSim significantly increases runtime, the automated generation tool provides a possibility to restrict the area within the overall study region in which traffic signal and lane data are created. In this application, a bounding box containing the inner city was chosen (see left part of Fig. 7). The so created scenario contains 12384 lanes on 4027 links and 7276 signals representing 1309 signalized intersections.

The approach presented in this paper provides a MATSim network and traffic signal and lane data for arbitrary regions. When generating MATSim scenario for a new region, the approach can be applied as a replacement of the standard network generation tool. If the approach is to be used to extend am existing MATSim scenario with traffic signal and lane data, the previously converged daily travel plans need to be cleared from routes and links as those information will not be valid anymore as network structure and IDs of links and nodes are different in the new network. Therefore, plans need to converge again within the co-evolutionary part of the simulation (see Sect. 2.2). Activities and coordinates can be kept.

For the present study one iteration of the co-evolutionary algorithm was run to verify that the automated generation of signal and lane data results in a connected and correct network, lanes are linked correctly to their links and upstream directions, and signals are properly positioned and grouped. For a subsequent application study it would be very interesting to run the extended Scenario until convergence for different signal control schemes and analyze their effect on the traffic flow dynamics, travel times, mode share etc. Before, one should preferably extend the current implementation of fast routing algorithms within MATSim to be usable with lanes. So far, only MATSim's Dijkstra implementation works with lanes, which is comparably slow.

5. Challenges

This section discusses the main challenges of the present approach. Some of them have been clear already during the implementation; others have been found when applying the approach to the Open Berlin Scenario (see previous section); and most of them could be solved by further extending the current implementation in a subsequent study.

Incomplete data. As the creation of the lane and signal information of this approach is based on OSM data solely, the quality of the resulting scenario is highly dependent on the completeness and correctness of the data provided by OSM. Fig. 8 shows today's availability of the three main tags used here: *lanes* for the number of lanes on a link, *turn:lanes* for the information about lane directions and *highway=traffic_signals* for the positions of traffic signals. Intuitively, the *lanes* tag (available since 2008) is most frequently used. Lanes are commonly used around the world

Fig. 7. Different network extracts of the Open Berlin Scenario with traffic signals and lanes. Left: City of Berlin and all generated signal locations (lying within a bounding box including the inner city); Middle: Region around the zoological garden – exemplary shows the correctness of signal locations; Right: Signalized roundabout of Großer Stern, visualizing one of the issues of the approach discussed in Sect. 5.

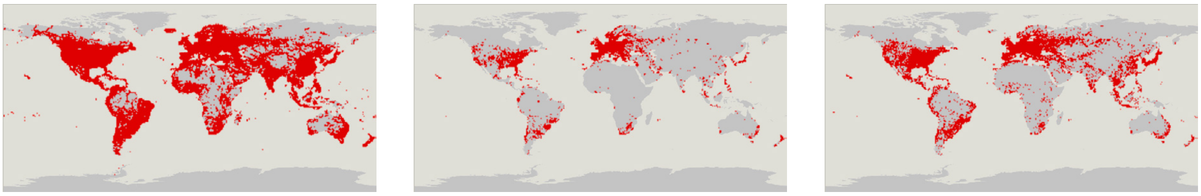

Fig. 8. From left to right: Usage of the tags *lanes*, *turn:lanes* and *highway=traffic_signals* on OSM [6].

and the number of lanes is an information that is easy to provide. Contrary to this, the *turn:lanes* tag providing the turn directions of lanes (available since 2013) is more complex to understand and fill and, therefore, less frequently used. The location of traffic lights seems to be tagged quite comprehensively given that they are not implemented everywhere around the world. However, before applying the presented approach to a new region, one should check and possibly correct and/or complete OSM data of that region first.

Nevertheless, the presented approach is also able to compensate for some missing data:

- A missing traffic light (*highway=traffic_signals*) is compensated if at least one signal per intersection is tagged.
- If no *lanes* tag is given, the number of lanes is guessed based on the *highway* type, i.e. the road category.
- For the compensation of missing turn directions of lanes (*turn:lanes*), the leftmost lane is treated as left turn only, the rightmost lane as straight and right turn, and (if existent) all other (middle) lanes as straight only.

Incorrect data. Unknown tags, which cannot be reasonably interpreted, are seldom, but exist in OSM. For example, tags like *none*, *yes*, *no* or an empty string occur as turn directions of lanes. All lanes with unknown *turn:lanes* tag are treated as straight and right turn. Besides wrong tags, also inconsistent data can occur: The presented approach ignores restrictions in case a link of an intersection would not have any out-links or no in-links. Otherwise, the network would not be connected.

Roundabouts. Angles between in- and out-going links at T-junctions of roundabouts behave differently than for regular intersections: A straight lane on a link l at a roundabout intuitively means staying on the roundabout. Contrarily, the angle between link l and its next link on the roundabout might suggest a (slight) left turn, especially for large roundabouts, whereas link l and the link that leaves the roundabout might form an angle closer to 180 degrees. Thus, the automatic assignment of to-links to lanes would lead to significantly wrong lane data. To compensate for this, all turns are allowed when the angles do not suggest clear directions, e.g. at large roundabouts. In the future, one could try to automatically identify roundabouts to further improve the flow values there.

The special case of signalized roundabouts constitutes another challenge within the current implementation, as the traffic lights of adjacent incoming approaches usually have a relatively small distance which might fall under the threshold for which signal data of adjacent nodes is merged into one intersection (see right case of Fig. 7). The suggested pre-identification of roundabouts would also help to solve this issue.

Five or more armed intersections. For such complex intersections, grouping oncoming traffic to the same signal phase is not trivial, as the identification of oncoming link pairs is not. The current approach, therefore, creates a single signal group for all incoming links of the intersection. This signal group, i.e. all signals, show green together for half of the cycle time. With this, the vehicles drive trough each other in MATSim. But at least, the green split (and with this average flow rates) are a good approximation. Alternatively, one could create a signal group per incoming link, which would impair the average flow rates of the fixed-time schedule but, on the other hand, improve the usability for adaptive signal control algorithms.

6. Conclusion

This study has presented the implementation of an automatic generation of traffic signals and lanes information for the agent-based transport simulation MATSim, solely based on OSM data. The usability of this method has been validated by applying it to the greater Berlin area. Besides improvements in terms of dealing with the challenges regarding large roundabouts and intersections with more than four incoming links (presented in the last section), future work should improve the implementation regarding: modelling of traffic signals for pedestrians, lane length and lane specific flow capacity values, e.g. based on the turn radius (as [1] suggests). Additionally, creating information about conflicting directions at signalized intersections would be very useful for the use of adaptive signal control methods in MATSim, which then would be able to adaptively group lanes into signal phases. However, OSM does not provide any information about which lanes are signalized separately. An extension of the traffic signal tags in OSM that is able to capture all this information has been suggested by Rieck et. al [7] but is not frequently used yet.

Acknowledgements

The authors thank Nils Schirrmacher for a systematic investigation of this topic in his master thesis at TU Berlin [8]. Funding by the Deutsche Forschungsgemeinschaft (DFG) for the project *Optimization and network wide analysis of traffic signal control* is gratefully acknowledged.

References

[1] FGSV, 2015. Handbuch für die Bemessung von Straßenverkehrsanlagen. Technical Report. FGSV.
[2] Grether, D., Bischoff, J., Nagel, K., 2011. Traffic-actuated signal control: Simulation of the user benefits in a big event real-world scenario, in: 2nd International Conference on Models and Technologies for ITS, Leuven, Belgium.
[3] Grether, D., Thunig, T., 2016. Traffic signals and lanes, in: [5]. chapter 12. doi:10.5334/baw.
[4] Grether, D.S., 2014. Extension of a Multi-Agent Transport Simulation for Traffic Signal Control and Air Transport Systems. Ph.D. thesis. TU Berlin. Berlin. arXiv:urn:nbn:de:kobv:83-opus4-53257.
[5] Horni, A., Nagel, K., Axhausen, K.W. (Eds.), 2016. The Multi-Agent Transport Simulation MATSim. Ubiquity, London. doi:10.5334/baw.
[6] OpenStreetMapFoundation, 2020. taginfo website. URL: https://taginfo.openstreetmap.org/tags. accessed: 2020-12-10 18:00 pm.
[7] Rieck, D., Schünemann, B., Radusch, I., 2015. Advanced traffic light information in openstreetmap for traffic simulations, in: Behrisch, M., Weber, M. (Eds.), Modeling Mobility with Open Data, Springer International Publishing, Cham. pp. 25–34.
[8] Schirrmacher, N., 2017. Open-Data-basierte Erstellung von Verkehrsnetzwerken zur Modellierung von Kreuzungen inklusive Lichtsignalanlagen und Fahrspuren. Master's thesis. Technische Universität Berlin. Also VSP WP 17-18, see http://www.vsp.tu-berlin.de/publications.
[9] Thunig, T., Kühnel, N., Nagel, K., 2019. Adaptive traffic signal control for real-world scenarios in agent-based transport simulations. Transportation Research Procedia 37, 481–488. doi:10.1016/j.trpro.2018.12.215.
[10] Ziemke, D., Kaddoura, I., Nagel, K., 2019. The MATSim Open Berlin Scenario: A multimodal agent-based transport simulation scenario based on synthetic demand modeling and open data. Procedia Computer Science 151, 870–877. doi:10.1016/j.procs.2019.04.120.
[11] Ziemke, D., Metzler, S., Nagel, K., 2018. Bicycle traffic and its interaction with motorized traffic in an agent-based transport simulation framework. Future Generation Computer Systems doi:10.1016/j.future.2018.11.005.
[12] Ziemke, T., Alegre, L.N., Bazzan, A.L.C., forthcoming. Reinforcement learning vs. rule-based adaptive traffic signal control: A Fourier basis linear function approximation for traffic signal control. Technical Report. Extension of ATT 2020 paper, submitted to special issue of AIC.
[13] Zilske, M., Neumann, A., Nagel, K., 2011. OpenStreetMap for traffic simulation, in: Schmidt, M., Gartner, G. (Eds.), 1st European State of the Map – OpenStreetMap conference, Vienna. pp. 126–134.