SUMO User Conference 2026

Scientific Paper

https://doi.org/10.52825/xxxx..... DOI placeholder (WILL BE FILLED IN BY TIB Open Publishing)

Published: (WILL BE FILLED IN BY TIB Open Publishing)

Reconstruction of Public Transport Routes Based on Imperfect GTFS and OpenStreetMap Data

An Analytical Approach to Improve Accuracy and Conversion to SUMO – Developed and Evaluated by an Example of the City of Magdeburg

Alexander Kaiser[1,*], and Alexander Schmaus[1]

[1]Otto von Guericke University Magdeburg, Germany

*Correspondence: Alexander Kaiser, alexander1.kaiser@ovgu.de

Abstract. A path-accurate simulation of individual bus, tram and train trips through a certain area requires accurate route data. If this is not given, e.g. as shape file from the public transport company, a reconstruction in SUMO is possible by means of the existing GTFS2PT Python tool, which imports and processes public GTFS and OSM data. However, the reconstruction of true routes often fails due to missing or outdated shapes in OSM, or due to inaccurate polylines in the GTFS shapes file, which often includes only stop locations and no further waypoints. Then GTFS2PT searches the fastest path between stops, which mostly results in the true route, particularly for a tram or train line (if alternative paths simply do not exist), but not in cases of bus lines within a dense road network. Therefor a map-matching approach based on GPS traces was recently proposed at the SUMO User Conference 2025 [1]. While these traces were given for Berlin in the corresponding GTFS shape file, this approach is not applicable to areas, e.g. Magdeburg, where they are not available in the local GTFS data. Hence, a more generic approach for bus and tram routes is required and proposed in this article, which is in line with GTFS2PT in terms of using open data sources (GTFS, OSM) and intends to improve the optimal path searching. This is achieved, for instance by limiting the set of potential edges for bus routes according to all edges used by other bus lines according to OpenStreetMap. Moreover, the overall fastest path is searched by considering all consecutive stops of a trip simultaneously. As the latter was already applied and tested successfully for tram lines in a previous article of the authors, it is adapted and tested more verbose for bus and tram lines in this article. Finally, the results of this approach are compared with the output of GTFS2PT considering all public transport trips within a selected day in the city of Magdeburg.

Keywords: Public Transport, Route Reconstruction, GTFS, OpenStreetMap

1 Introduction

With the growth of Intelligent Transportation Systems (ITS), such as on-demand mobility services, car-sharing, and autonomous vehicles and buses, the need for more sophisticated verification and validation methods is increasing. In the project IMIQ (Intelligent Mobility Space in the Quarter), ITS are tested in a real-life laboratory in the city of Magdeburg, Saxony-Anhalt, Germany [2]. These ITS require, among other things, precise traffic predictions. However, as conventional traffic forecasts cannot be applied in this context, a large-scale traffic simulation based on real-world open-source data is used instead to provide detailed traffic information.

We perform the simulation using SUMO [3]. The results are supposed to be used within an urban digital twin of the so-called Science Port in Magdeburg. Besides the establishment of the real-life laboratory, which is mentioned above and also located in the Science Port, the urban digital twin is supposed to be the main outcome of the IMIQ project. General and special approaches to combine SUMO with digital twins and 3D visualization tools, respectively, are described extensively in the literature, among others in [4], [5]. With respect to the IMIQ project, we already described a first concept to integrate microscopic traffic simulation in the digital twin. For more information, we refer to the introduction of our previous paper [6] at this point.

SUMO enables the microscopic traffic simulation over a total day, even in large-scale models of entire cities, while the computation time is still acceptable. Rather, the challenge lies in building these large-scale models with respect to sufficient accuracy and reasonable time effort. However, the automatization of building a SUMO model for a certain area is already supported graphically by the OSM Web Wizard tool, or programmatically by the combination of existing scripts from the SUMO tool collection (e.g. osmGet.py, osmBuild.py, netconvert). Examples of building large-scale models with SUMO are also presented extensively in the literature, among others for Montreal [7], Luxembourg [8] and Berlin [9].

While map data to construct the road and railway network of an arbitrary area is usually available from OpenStreetMap [10] in a sufficient manner, the acquisition of real traffic data to generate trips of different transport modes and vehicle classes is still a challenging and time-consuming task. As we focus on public transport in this article, we do not consider demand modelling of passenger cars, although this is already implemented in our SUMO model, but not published so far. To model the trips of public transport vehicles, which usually follow pre-defined routes with corresponding sequences of stops and schedules, the digital timetable data of the public transport service providers are required. For this purpose, we retrieve the data in the General Transit Feed Specification (GTFS) format [11], which is provided for different parts or even the entire area of Germany by open data platforms, particularly [12]. Besides, static map data like route shapes and stop locations are also retrieved from OpenStreetMap.

A key requirement of our simulation model is the accurate mapping of tram and bus routes in the city of Magdeburg. GTFS provides line definitions, stop sequences, schedules, and stop locations and is frequently updated. However, it represents the routes in Magdeburg only as ordered lists of stops and includes coordinates that may deviate from their actual locations (cf. **Figure 1** b). OSM, in contrast, typically offers accurate route geometries along streets or railways (cf. **Figure 1** a), but temporary changes (e.g. detours) are often missing or outdated. Consequently, neither dataset alone is sufficient to reconstruct realistic daily tram and bus routes. Only their combination enables both operational correctness and geometric accuracy.

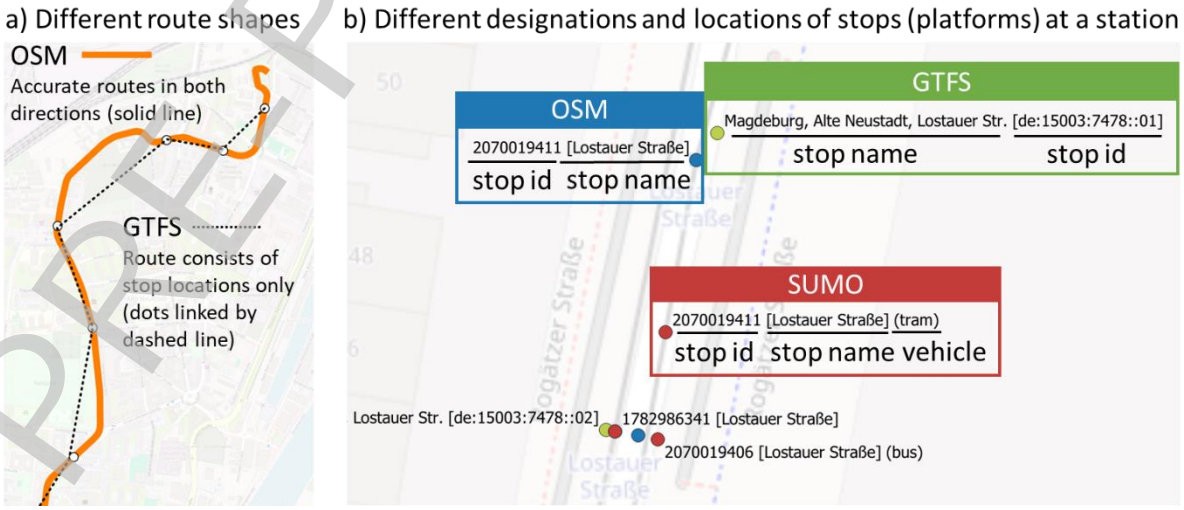

Figure 1. *Typical differences between GTFS, OSM and SUMO concerning routes and stops [6]*

The problem of merging different stop data from GTFS and OSM in order to find the correct locations (coordinates) and designations (id, name), is already analysed and solved in our previous paper [6]. Similar approaches for the stop location problem are described in [13], [14]. Since stop location data, which is required for analysing and solving the routing problem in this article, is derived from our own method, it is necessary to sum up this method as follows.

In the first stage, given stops are grouped by vehicle class (bus, tram, train) within each database (OSM, GTFS). In SUMO, a stop (busStop, trainStop) depicts a platform beside a street or track (lane) for busses, trams and trains, respectively. In reality, a station along a bus, tram or train line usually consists of two opposite stops, at least one stop for every direction, or multiple parallel stops for different lines in the same direction. These stops are named equally (e.g. “Magdeburg, Lostauer Straße” in **Figure 1** b). If the number of stops at a station given from the original data is insufficient, SUMO Netconvert will create additional stops automatically, as also shown in **Figure 1** b, to provide a stop for every vehicle class per direction. Finally, our method compares all stops of the model with respect to position, id and name, to identify and group the stops of every station. Thereby, similar stops are merged to clusters which represent the stations in reality. This is exemplified for different stations in **Figure 2**.

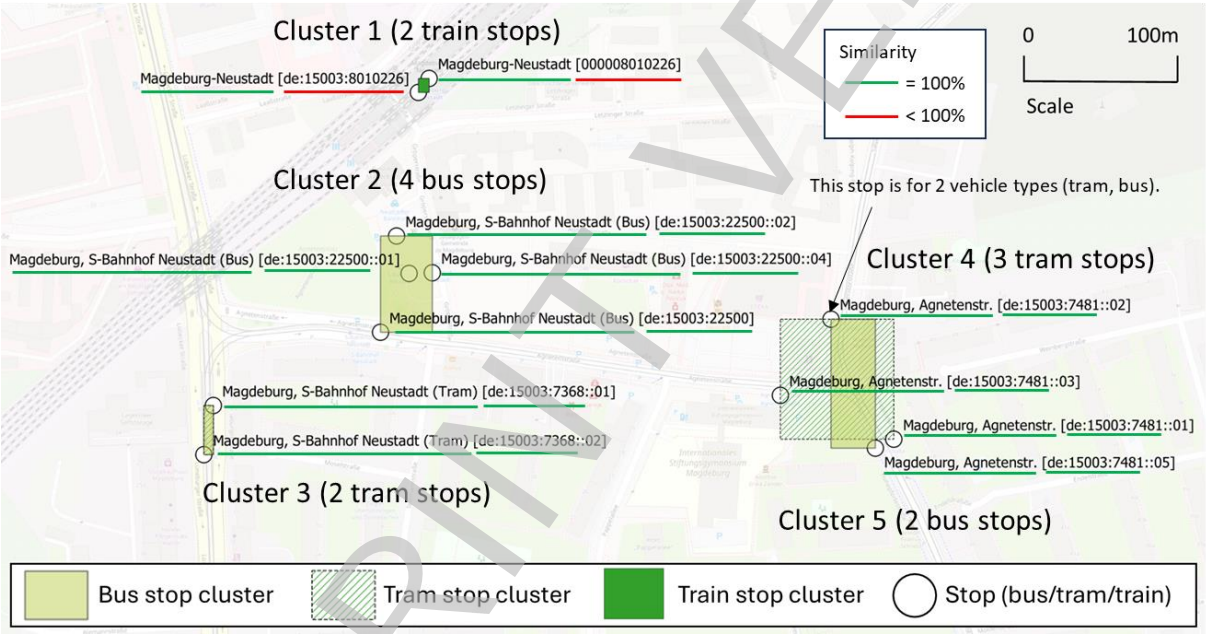

Figure 2. *Example of matching and clustering similar GTFS stops (dots) to a station (rectangles) [6]*

In the second stage of our stop location method, the stations (stop clusters) of GTFS and SUMO (including OSM) are compared analogously to single stops in the first stage, i.e. in terms of spatial distance and similar designation. The latter is performed by the names only, as the original identifiers are completely different in GTFS and OSM. As a result, every stop in the SUMO model is clearly allocated to a GTFS stop, both represented by stations (clusters). According to the GTFS stops sequence of every trip given from GTFS data, a path-accurate and trip-individual reconstruction of every route, including stops and departures, is possible and described in detail in this article.

The article is structured as follows. In Section 2 we describe our routing method in detail. In the first stage of this method, alternative routes between consecutive stops pairs derived from all trips are calculated (Section 2.1). In the second stage, these stop-to-stop routes are connected to a total route of every trip (Section 2.2). In Section 3 we present exemplary results of our routing method, which are also compared with results of the alternative GTFS2PT tool. Finally, our conclusions and a brief outlook on further research are carried out in Section 4.

2 Method

2.1 Determining Alternative Routes from Stop to Stop

In the first step, the existence of a directional connection between two stations (GTFS cluster) is determined, which results from at least one bus or tram trip in the available GTFS timetable data. All resulting connections between stations are recorded in a matrix of dimension (n,n) by the value "1" (connection from i to j exists), where n corresponds to the number of stations (GTFS cluster) in the model. This is illustrated by the example of a tram network with 25 selected stations in **Figure 3**.

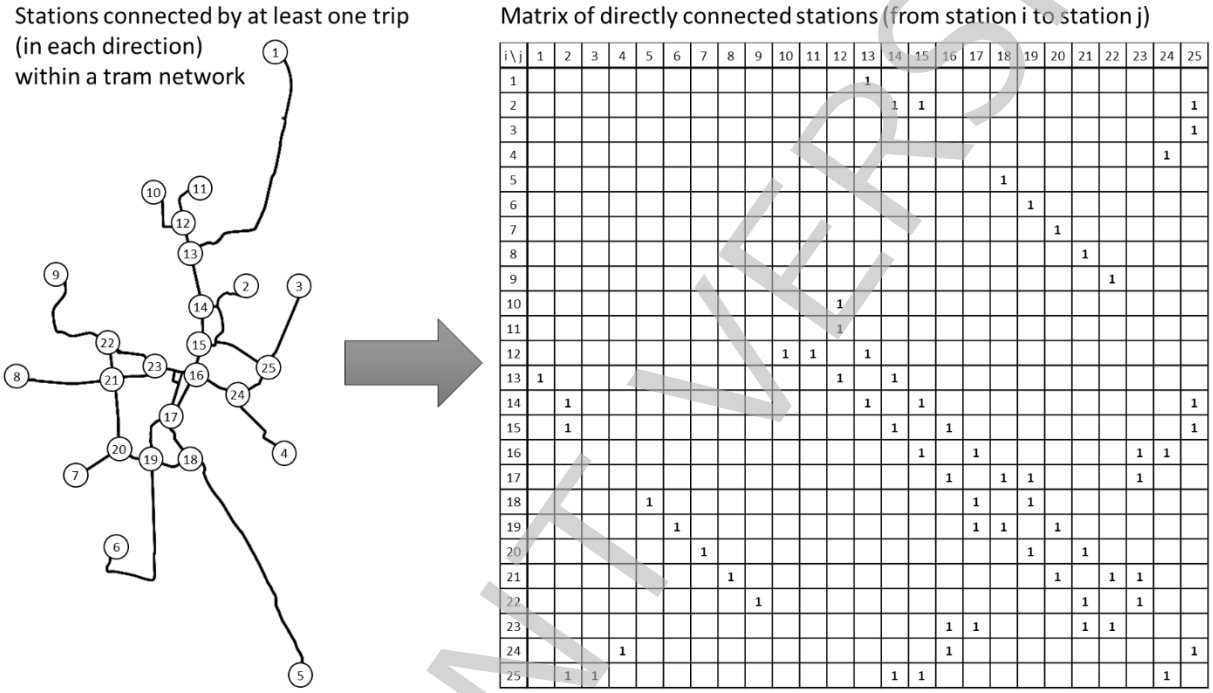

Figure 3. *Example of building a connection matrix of a tram network with 25 selected stations*

In the second step, the different connection options of the bus or train platforms (stops) between two stations (GTFS cluster) are determined. The positions and orientations of the bus or train platforms (stops) in the SUMO model are decisive. If there are several stops within a cluster, all combinations between the stops of cluster i and cluster j (i.e. n_i x n_j connections) must be taken into account, since the directionality of a stop, i.e. whether it is intended for travel from i to j or from j to i, is unknown. However, this is not absolutely necessary for the procedure and arises implicitly later after the entire route has been determined (Section 3). In addition, there may be several stops in each direction, e.g. for parallel bus or train lines, which are also taken into account by the procedure. The principle described is schematically illustrated in **Figure 4** using the example of two consecutive stations of a bus line. Station 1 consists of two stops (one stop in each direction). Station 2, on the other hand, consists of four stops, as several bus lines run from here in different directions.

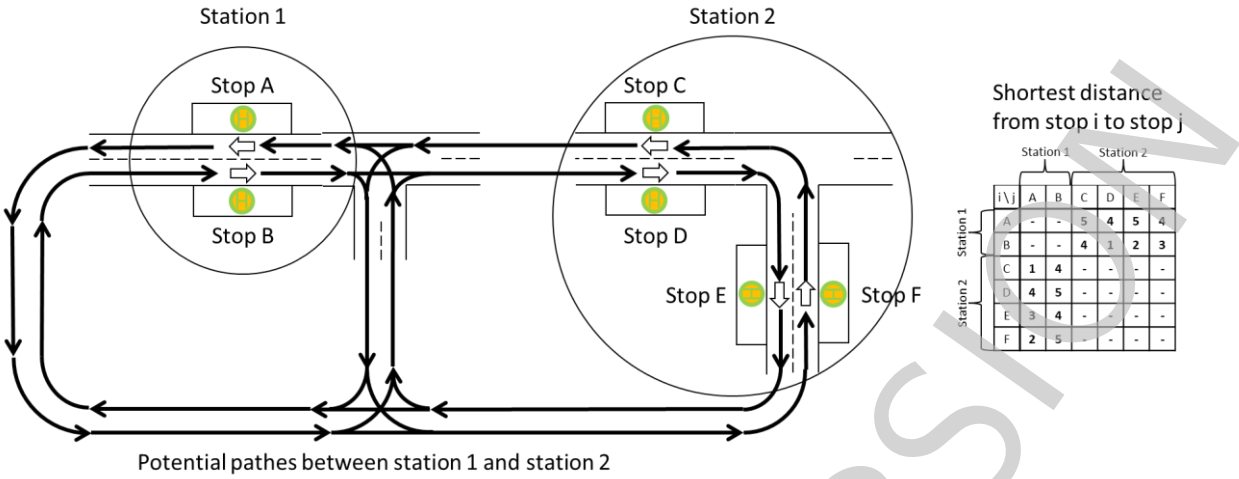

Figure 4. *Example of potential paths between 2 bus stations, including 2 and 4 stops, respectively*

As also symbolised by the small matrix in **Figure 4** on the right, the next step is to determine the optimal route to each connection option. For this purpose, SUMO's internal optimization tool Duarouter is used, which receives the respective section of the path (edge) where a stop is located and the corresponding vehicle class as input. In order to take into account any information available on the regular routing of a line from OpenStreetMap in the best possible way, the Duarouter also applies the corresponding route sections (edges) as intermediate destinations, i.e. they are allocated to the “via” attribute (not shown in **Figure 4**). However, the optimal route without intermediate destinations is always calculated, as the intended regular route (according to OpenStreetMap) may not be permitted in the SUMO model. Thus, up to two permissible routes can exist between two stops, of which the one intended for the respective bus or train line is preferred when the total route is later determined (section 4), even if it exhibits a longer travel time.

While the shortest permissible route between two stops for trams usually corresponds to the actual route due to a thin rail network and a few turning options (e.g. turning loops at terminal stops), in case of buses further restrictions of optimization have to be taken into account as follows. First, all edges of the road network, where no bus runs according to OpenStreetMap, are rated with a relatively high weight (here with a travel time of 3600 seconds per edge, which results in a speed close to zero for edges mostly short in the urban area). As shown in the example of our SUMO network model (**Figure 5**), all gray edges are weighted with this high value of travel time, while the actual value of travel time is maintained on the remaining red edges (preferred bus network). As a result of optimization, gray edges are only selected, if there is no alternative route between two stops on red edges only. Secondly, U-turns which are rather atypical for buses (except in the area of terminal stops) are also punished by an additional travel time (e.g. 3600 seconds) in order to avoid them. Consequently, these two restrictions make it possible to reconstruct realistic bus routes in a dense urban road network even without specifying the actual routes of bus lines.

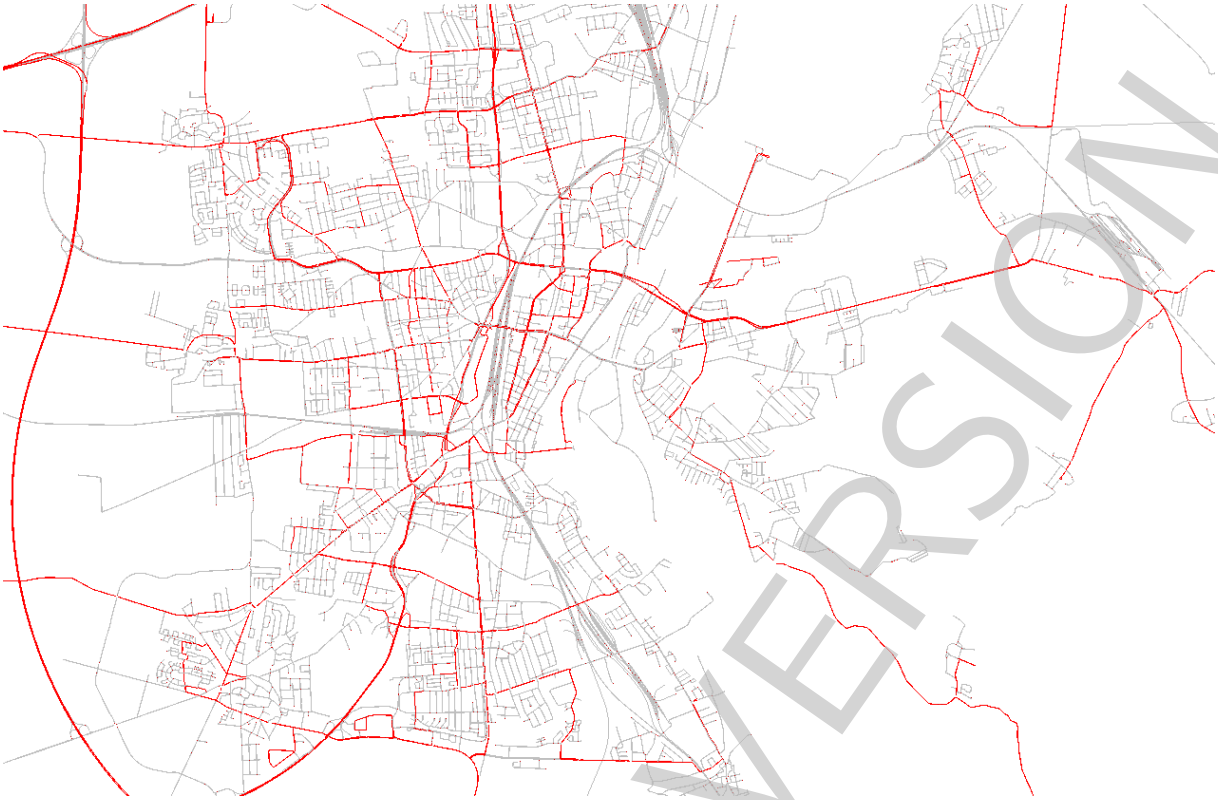

Figure 5. *Edges preferred for bus trips (red lines) in an exemplary SUMO net model*

2.2 Determining the Total Route of Individual Trips

The partial routes between two consecutive stations defined in the previous section correspond to the segments of the total routes of all bus or tram trips. In order to select the "true" partial route among alternative connection options (and thus implicitly also the "true" stop within a cluster), this method assumes that the merging of all selected partial routes leads to the shortest total path according to the sequence of stops of a trip. As a solution to the optimization problem, the Dijkstra algorithm is applied analogously to the Duarouter. However, since the elements of the corresponding graph do not correspond to the edges of the SUMO network model, but to the alternative connection options between the stations (partial routes), the Dijkstra algorithm is to be implemented in an individual program, here with the help of the Python library SciPy. The relevant steps of this program, which also includes the preparation of input data for optimization, are described in the following.

Figure 6 illustrates the procedure using the example of a bus trip section including four stations, where each station is named with a number for simplification purposes (cf. **Figure 6** a, b and c). In the first step, the sequence of stations for each trip of a bus line, which is illustrated as linearised route in **Figure 6** c, is read from the GTFS timetable data. Within a line with outward and return direction, there are at least two station sequences for all trips, as both directions of travel are to be considered separately. If the sequences of a line differ in at least one station or the start or end are shortened (i.e. a terminus of the trip is brought forward), these are considered new, additional routes. The advantage of this trip-by-trip view of a line is that routes that deviate from the normal case, e.g. temporary detours due to road closures or trips to or from the depot at the beginning or end of the day, are also taken into account. In OpenStreetMap, these different routes are usually not implemented or updated, while in the GTFS data all trips, even with different stop sequences, are usually given. However, the corresponding GTFS route shapes (geo-coordinates) are usually not path-accurate and only given for stations, as already mentioned above.

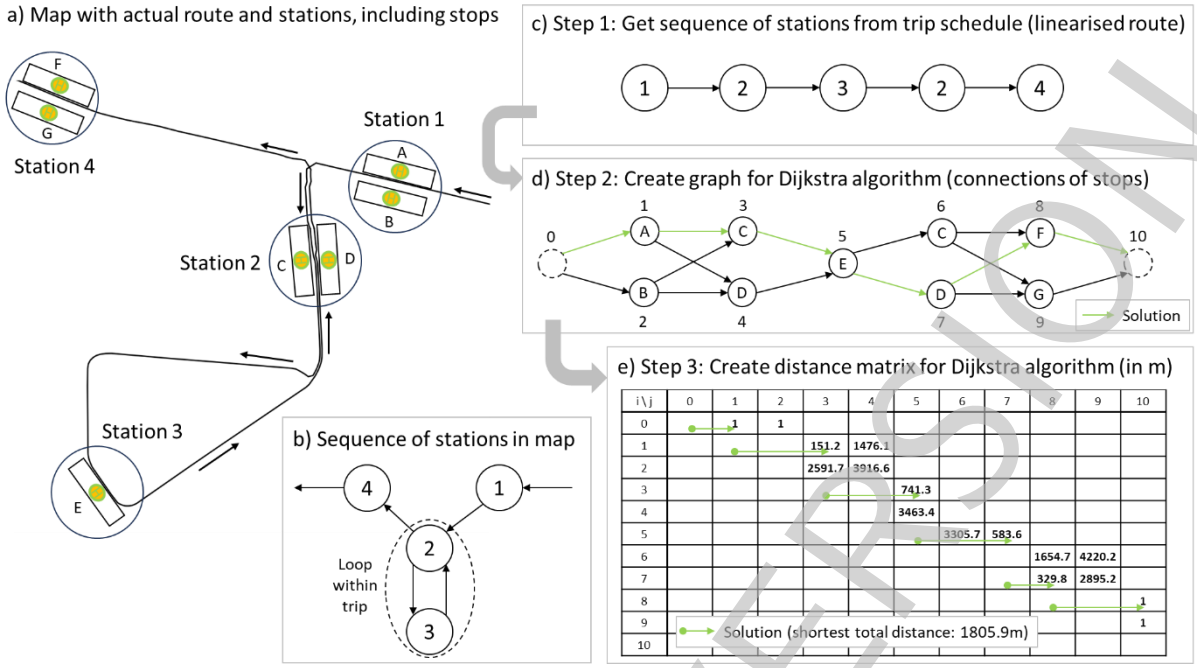

Figure 6. *Steps to determine a total route – exemplified by four stations of a bus trip, including a loop*

In the next steps (2 and 3), all stops from all stations of a trip are listed in the corresponding order of the station sequence. This list forms the basis for the numbering of the nodes of the graph or the rows and columns of the matrix for the Dijkstra algorithm (cf. **Figure 6** d and e). If individual stations are served twice (as applicable to station 2 in our example) or even more (i.e. n-fold) during a trip, they have to be listed several times in the list, otherwise the Dijkstra algorithm would exclude these loops in order to get to the destination node by the shortest path. The weights of the edges in the graph correspond to the actual lengths of the partial routes (i.e., from stop to stop, not from edge to edge). In this case, a connection which is intended for the respective bus line according to OSM data is not weighted with the actual distance, but a minimum value (e.g. 1), so that the Dijkstra algorithm selects it preferentially. At this point, it is also checked whether the planned partial route passes at least one other station of the current trip. If this is true, the routing between the two stations would not be efficient without a stop at this intermediate station and would be also rather strange in practice. It is therefore considered an incorrect assignment and excluded. Instead, the shortest route between the two stops is used (i.e. without intermediate destinations, but still within the network intended for buses) and weighted with its actual distance in the graph and matrix.

In the last, optional step, all individual points of the edges are converted into geo-coordinates and connected as a polyline to check every route projected onto a map. This visual check by the user is recommended, because the Duarouter algorithm, which was used to find the stop-to-stop routes in previous section (2), sometimes created errors like redundant edges, particularly in roundabouts, resulting in multiple loops. Since the latter occurred as a systematic error during our tests and we were not able to fix this error in the Duarouter itself, we implemented a check and correction function in our programme to repair routes automatically (i.e. redundant edges are removed) before the visual check.

At the end of the procedure, a SUMO-specific route file (*.rou.xml) for all bus or tram trips on a specific day can be created automatically according to the GTFS timetable data. If this file still contains errors in the routes or schedules, these are usually automatically detected right at the beginning after loading the file in the SUMO simulation. In this case, the simulation cannot be started until the errors are corrected by the user. However, the tests carried out so far with data for the area of the city of Magdeburg, which are presented in Section 4, have not found any errors that needed to be corrected manually in the corresponding route files.

3 Results

In this section, the functionality and solution quality of our new method is demonstrated with exemplary results. These results are calculated again on the basis of map and timetable data for the area of the city of Magdeburg, analogous to the examples in previous sections. To assess the solution quality, the final output (route file) of our new method is compared to the respective output of the existing SUMO tool GTFS2PT. This approach corresponds to a comparative assessment, which is an alternative or additional option to the comparison with ground-truth data. The latter are hardly available from a third independent database, because GTFS and OSM are already used and each is not completely correct to serve as ground-truth. In our case, we would have to collect primary data of bus and tram routes and schedules by extensive observations in the city of Magdeburg. Consequently, only the comparative assessment approach is feasible with reasonable effort in the current state regarding data availability. In cases of doubt, the correctness of individual results could so far only be checked by the author on a random basis with the help of Internet research and observations on site as well as the author's deep knowledge as lifelong inhabitant of the city of Magdeburg.

This chapter is structured as follows. For the sake of clarity and reproducibility, the setups of both methods, particularly GTFS2PT, are described in next Section 3.1. After that, the results are presented on an aggregated level regarding trip quantities and network usage in Section 3.2, as well as on a detailed level in Section 3.3 by comparing two trips with identical IDs, which may differ in terms of route, stops and departures calculated by each method. Finally, the computing times of both methods are compared in Section 3.4, and a brief summary of all results is delivered in Section **Fehler! Verweisquelle konnte nicht gefunden werden.**.

3.1 Setup

The GTFS2PT method we used for our experiments is available as a Python script (gtfs2pt.py) in SUMO version 1.26.0 from 29.01.2026. To run the script in the case of buses (trams analogously), the following command according to [15] was applied in a terminal window:

python *tools/import/gtfs/gtfs2pt.py* ***--network*** *osm_md_road+rail_r2.net.xml* ***--osm-routes*** *osm_sa_total_ptlines.xml* ***--stops*** *osm_md_road+rail_ptstops.add.xml* ***--gtfs*** *20260126_opendata_md.zip* ***--date*** *20260205* ***--modes*** *bus* ***--sort --repair --vtype-output*** *gtfs_pt_vtypes.xml*

The command line begins with the call of the Python script, i.e. the relative file path, followed by relevant options, which partly trigger algorithm 1 and 2, respectively, and are explained in **Table 1**.

Table 1*. Description of relevant options of gtfs2pt.py*

	Option	Description (quoted from original Python script)	A1	A2
1	--network	sumo network to use	x	x
2	--osm-routes	osm routes file	x	
3	--stops	files with candidate stops (selected by proximity)	(x)	x
4	--gtfs	define gtfs zip file to load (mandatory)	x	x
5	--date	define the day to import, format: 'YYYYMMDD'	x	x
6	--modes	comma separated list of modes to import (bus, train, tram, light_rail, monorail, subway, aerialway, ferry)		
7	--sort	sorting the output-file		
8	--repair	repair osm routes		
9	--vtype-output	file to write the generated vehicle types to		
Abbreviations: A1…algorithm 1 A2…algorithm 2 x…mandatory (x)…no impact				

Regarding the second and third option (--osm-routes and --stops), it is worth mentioning that GTFS2PT consists of two different algorithms for stop localisation, which are triggered either by --osm-routes (algorithm 1) or --stops (algorithm 2). The first algorithm maps nearby stops to the OSM route edges given from the corresponding xml file (e.g. osm_sa_total_ptlines.xml), and ignores potentially locations given from OSM, even if these are explicitly entered with the --stop argument. However, every stop location given from the respective GTFS file, which is a mandatory input, is considered to find the closest feasible location. If, on the other hand, algorithm 2 is triggered by --stops (without --osm-routes), potential stop locations are explicitly given from file as candidates (e.g. osm_md_road+rail_ptstops.add.xml). Therefore, algorithm 2 usually provides stop locations which should be always in line with OSM, while the stop locations of algorithm 1 should be closer to GTFS. However, we were unfortunately not able to run the second algorithm without any errors, to verify our assumptions and to get stop locations in line with OSM, whose static data is more reliable than GTFS (as explained in Section 1). Since all results of GTFS2PT are based on algorithm 1, potential deviations of stop locations in comparison to our new method are expected in the following.

The setup of our new method is already described in the Section 3 and therefore needs no further explanation here, except in terms of input data applied for the simulation scenario. The input of both methods is OSM data current at the time of the study (download of the OSM file for the study area on 19.01.2026) and the GTFS timetable data (download of the ZIP file for Germany from 26.01.2026). Out of this, a timely, busy weekday, in this case Thursday, 05.02.2026, was selected as simulation scenario to run both methods and compare the results. Although there were school holidays on this day, there was no holiday timetable with a reduced number of trips on most bus and tram lines.

For the sake of better clarity and verifiability of the present results, only those bus and tram lines whose termini are in the urban area of Magdeburg will be selected for the following considerations. This means that all regional buses are excluded and all tram and bus lines of the local public transport company, Magdeburger Verkehrsbetriebe (MVB), are included.

3.2 Trip Quantities and Network Usage

In the first step of data analysis on an aggregated level, all trips on the day of investigation are extracted from the respective route files and evaluated in terms of numbers (quantities). For this purpose, the number of trips per line and the corresponding difference from the target value, i.e. the number of trips planned according to the GTFS data, is calculated (see **Table 2** for buses and **Table 3** for trams). If there is no difference to the target value (i.e. $\Delta = 0$, which is indicated in simplified terms by an empty cell) and the number of trips is the same for both methods, this indicates that the data in both methods is correct. However, errors can still be detected when looking at individual trips in detail with regard to the spatial and temporal course (i.e. routing, station sequence and departure times), which is why a detailed analysis of the trips will be carried out in next section (3.3).

With our new method, only 2 out of a total of 20 bus routes have different numbers of trips (see rows 5 and 19 in **Table 2**). With the GTFS2PT method, there are only differences in the night bus lines, which are also sometimes very high, i.e. by up to -100 percent compared to the target value (see rows 15 to 20 in **Table 2**). Overall, a preliminary correctness can be determined on the basis of the number of trips for 13 bus lines. These include 1221 of a total of 1360 trips (89.8%).

***Table 2**. Number of bus trips planned (target value from GTFS data) in comparison with the number of trips generated (actual value) by our new method respectively the existing GTFS2PT tool*

	Bus Line (ID from GTFS)	Number of Trips				
		Planned	New Method (Δ)		GTFS2PT (Δ)	
1	de:nasa:15003|Bus|51:MVB_3	95	95		95	
2	de:nasa:15003|Bus|52:MVB_3	106	106		106	
3	de:nasa:15003|Bus|53:MVB_3	95	95		95	
4	de:nasa:15003|Bus|54:MVB_3	115	115		115	
5	de:nasa:15003|Bus|56:MVB_3	42	40	(-2)	42	
6	de:nasa:15003|Bus|57:MVB_3	102	102		102	
7	de:nasa:15003|Bus|58:MVB_3	110	110		110	
8	de:nasa:15003|Bus|59:MVB_3	11	11		11	
9	de:nasa:15003|Bus|61:MVB_3	45	45		45	
10	de:nasa:15003|Bus|66:MVB_3	37	37		37	
11	de:nasa:15003|Bus|69:MVB_3	122	122		122	
12	de:nasa:15003|Bus|71:MVB_3	146	146		146	
13	de:nasa:15003|Bus|72:MVB_3	110	110		110	
14	de:nasa:15003|Bus|73:MVB_3	127	127		127	
15	de:nasa:15003|Bus|N3:MVB/83_3	16	16		0	(-16)
16	de:nasa:15003|Bus|N4:MVB/84_3	16	16		9	(-7)
17	de:nasa:15003|Bus|N5:MVB/85_3	17	17		0	(-17)
18	de:nasa:15003|Bus|N6:MVB/86_3	16	16		9	(-7)
19	de:nasa:15003|Bus|N7:MVB/87_3	18	10	(-8)	20	(+2)
20	de:nasa:15003|Bus|N9:MVB/89_3	14	14		15	(+1)
	Total	1360	1350	(-10)	1316	(-44)

In the case of tram lines, there are clear differences between the two methods. With our new method, there are no differences from the target values, while with the GTFS2PT method this applies to only one of 10 lines (see row 2 in **Table 3**). All other lines show high deviations of up to -100 percent (see lines 1 and 8 in **Table 3**). In this respect, only tram line 2 with 190 trips can be noted as provisionally correct. This corresponds to a share of only 14.0 percent.

***Table 3**. Number of tram trips planned (target value from GTFS data) in comparison with the number of trips generated (actual value) by our new method respectively the existing GTFS2PT tool*

	Tram Line (ID from GTFS)	Number of Trips				
		Planned	New Method (Δ)		GTFS2PT (Δ)	
1	de:nasa:15003|Strab|1:MVB_0	204	204		0	(-204)
2	de:nasa:15003|Strab|2:MVB_0	190	190		190	
3	de:nasa:15003|Strab|4:MVB_0	201	201		190	(-11)
4	de:nasa:15003|Strab|5:MVB_0	120	120		113	(-7)
5	de:nasa:15003|Strab|6:MVB_0	198	198		191	(-7)
6	de:nasa:15003|Strab|9:MVB_0	204	204		198	(-12)
7	de:nasa:15003|Strab|10:MVB_0	192	192		93	(-99)
8	de:nasa:15003|Strab|N1:MVB/81_3	16	16		0	(-16)
9	de:nasa:15003|Strab|N2:MVB/92_0	16	16		18	(+2)
10	de:nasa:15003|Strab|N8:MVB/98_0	16	16		18	(+2)
	Total	1357	1357		1011	(-346)

However, line N1 (row 8 in **Table 3**) is actually a night bus, which is incorrectly named as tram (in German abbreviated "Strab") in its ID, whereas the route type (3 = Bus) is correct in the respective file of the GTFS dataset (routes.txt). Since our new method has so far always

been based on consistent GTFS data regarding the vehicle type defined in two different parameters (1. route_id, 2. route_type), in this case the route of line N1 was modified by slightly relocating two tram stops, in order to run all trips by tram (according to the so far preferred route_id) in the simulation model. Nevertheless, our new method still needs to be improved at this point in order to automatically detect and correct erroneous or inconsistent GTFS data (e.g. by matching the vehicle type between the line and the associated stations of the trips).

In the second step, the routes of all bus and tram trips are projected onto a map (see **Figure 7** for buses and **Figure 8** for trams). The advantage over the purely numerical comparison is that possible differences in route guidance can be seen at a glance. For better clarity, buses and trams are displayed separately and night bus lines (N3 – N7 and N9) are excluded, as these are almost completely missing in the GTFS2PT method (cf. **Table 2**) and are therefore not comparable with the routes of our new method. However, when comparing the left and right sides of the map – due to the continued aggregated approach – only those roads or railways can be identified on which at least one trip of a bus or tram line takes place. Which specific trips or lines are different and possibly faulty can only be determined during the detailed analysis in the next section (3.3).

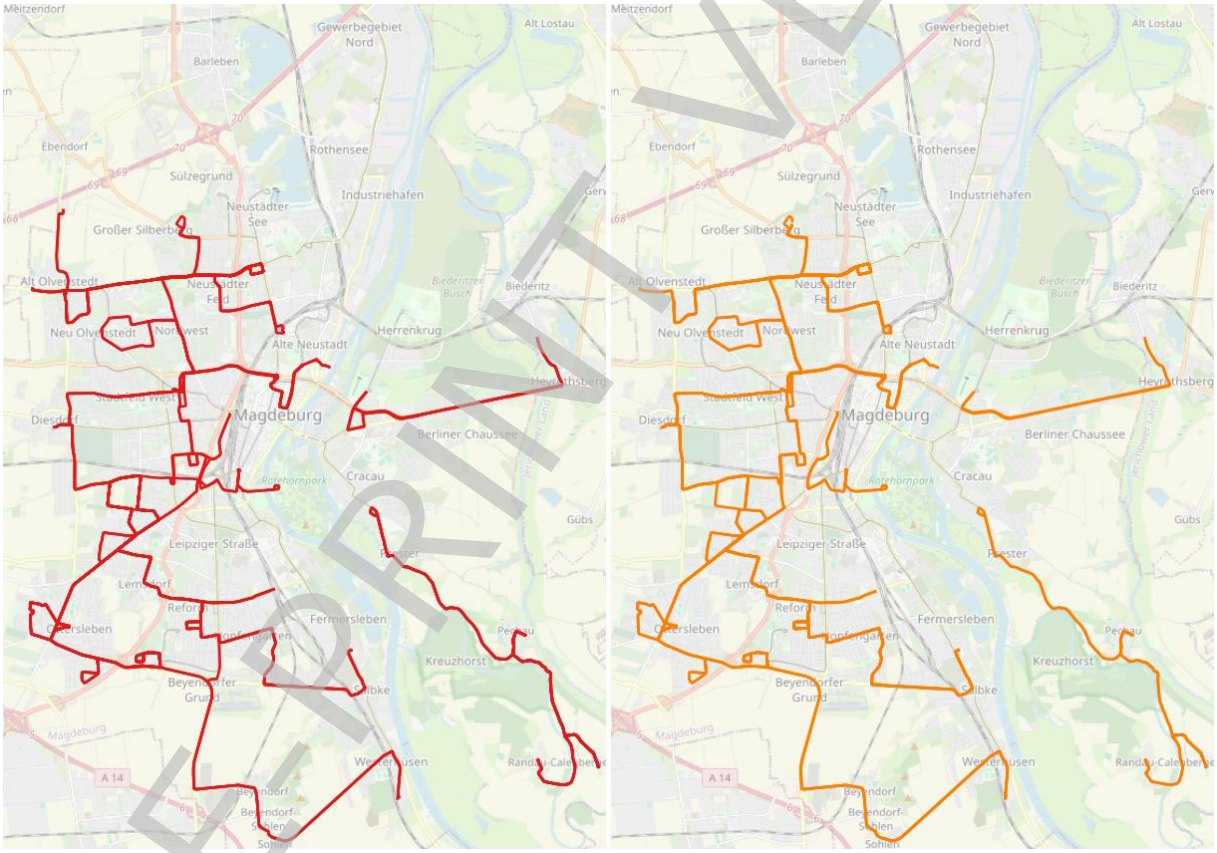

Figure 7. *Routes of bus lines in Magdeburg on 5th February 2026, generated by our new method (red lines, left) in comparison with the routes generated by GTFS2PT (orange lines, right). All regional and night buses are excluded. Differences between bus networks are highlighted in* ***Figure 9*** *a-b.*

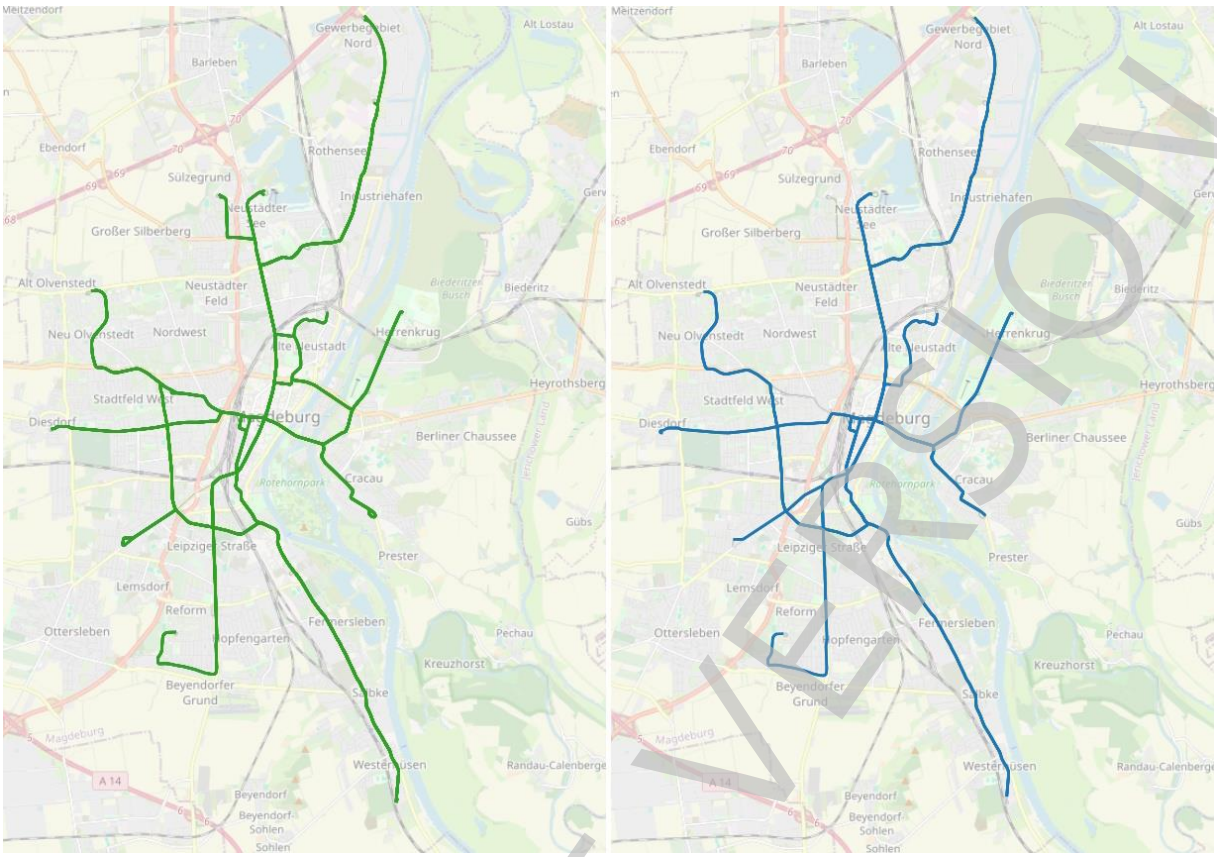

Figure 8. *Routes of tram lines in Magdeburg generated by our new method (green lines, left) in comparison with the routes generated by GTFS2PT (blue lines, right). Differences between tram networks are highlighted in* ***Figure 9*** *c-d.*

The different routes are represented by superimposing the line networks from both methods and highlighting the corresponding points in colour in **Figure 9**. In part a (left), the bus route network of our new method (grey) is superimposed on the network of the GTFS2PT method (orange), the latter being recognisable at one point (no. 1) by the orange line (this is the street called Südring in Magdeburg). This means that a route section may be missing at this point in our new method. Analogously, the reverse difference is shown in part b (middle-left), with the corresponding differences being indicated by red lines. These are four different points in the route network according to the GTFS2PT method (short differences at some termini are neglected here and will be examined in next section). The position 1 in part a shows the regular course of a bus line (line 52 via Südring), which is currently diverted due to a construction site of several months, which is only correctly indicated in the GTFS data, but not in the OSM data. The diversion route is shown by position 3 in part b (line 52 via the stop "Jordanstraße" in Halberstädter Straße). However, additional edges had to be added to the bus network in our new method, as these were not previously part of the regular bus network.

In the case of the tram network, according to the GTFS2PT method in part d (**Figure 9**, right), four different positions can also be seen. For example, position 2 involves morning and evening depot trips through a street (Agnetenstraße in Magdeburg) that is otherwise not regularly used by trams. The route is therefore not included in the OSM data. Position 4 results from a no longer existing, large-scale diversion of tram lines due to a road closure (at Damaschkeplatz in Magdeburg between spring and autumn 2025) and is therefore also an error. The other positions (1 and 3) are just as flawed according to on-site research, but have not yet been able to be checked with regard to the cause. In addition, in part c (middle-right), a missing passage is marked in blue by a street that is temporarily closed to trams (Halberstädter Straße between Südring and Leipziger Straße). This is an error in the GTFS2PT method caused by incorrect OSM data which have not been updated accordingly.

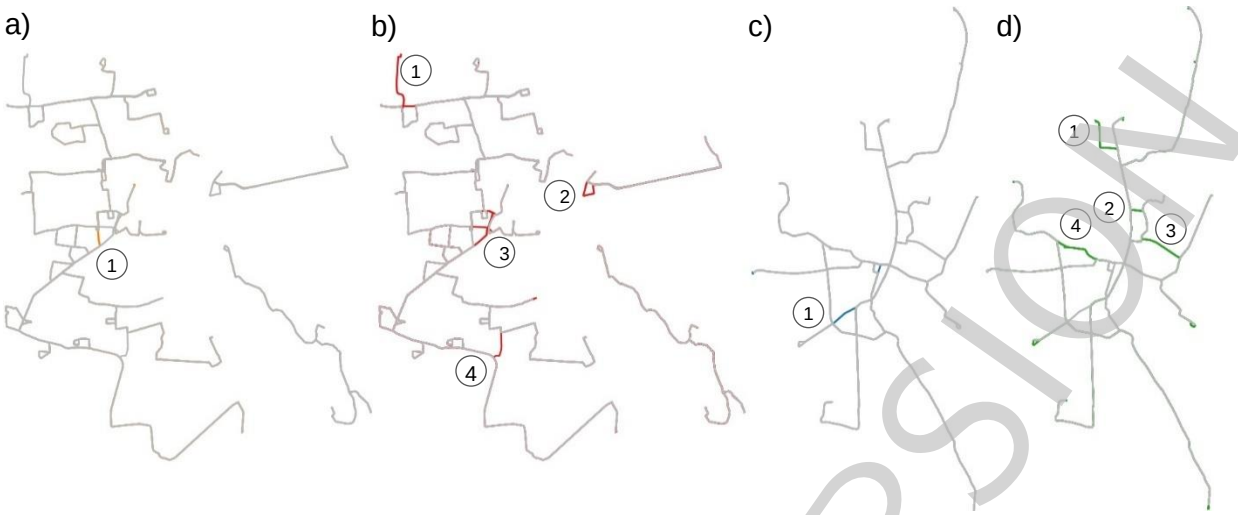

Figure 9. *Differences between network usage by buses (a-b) and trams (c-d): a) orange sections are additionally used by buses according to GTFS2PT, b) red sections are additionally used by buses according to our new method (NM), c) blue sections are additionally used by trams according to GTFS2PT, d) green sections are additionally used by trams according to NM*

3.3 Comparison of Route, Stops and Departures by Trip

In the detailed result data analysis, the pairs of comparable trips from both route files (our new method and GTFS2PT) are first identified by matching all trip IDs. Two trips with identical IDs are then checked for correspondence with regard to the characteristics of route (sequence of edges), sequence of stations and departure times at the stations (sequence of departures) using appropriate metrics.

Due to the trip-by-trip comparison based on the trip ID, all trips within Magdeburg including night buses can be taken into account. Overall, the quantities of trips listed in **Table 4** are included in the corresponding route files (rows 1 and 2 respectively). The planned trips according to GTFS are given in row 3 and correspond to the sums in **Table 2** and **Table 3**, respectively. The remaining intersections of comparable trips (row 4) are analysed in the following.

Table 4*. Number of trips to be compared out of the total number of trips in the route files*

	Number of Trips …	**Buses**	**Trams**
1	Route file of new method (all lines)	1372	1357
2	Route file of GTFS2PT (all lines)	1770	1011
3	Trips selected (MVB lines only)	1360	1357
4	Trips comparable (MVB lines only)	1301	1007

The equality or difference of a route (sequence of edges) is determined by the metric Identical Sequence of Edges (ISE) according to Equation 1. Here, index 1 (trip 1) refers to our new method and index 2 (trip 2) to the GTFS2PT method.

$$\text{Identical Sequence of Edges (ISE)} = \begin{cases} 1, \text{if sequences of edges in trip 1 and 2 are identical} \\ 0, \text{else} \end{cases} \quad (1)$$

In order to calculate the metrics for the equality or difference of the sequence of stations (Eq. 3) and sequence of departure times (Eq. 4), the same number of stations is assumed (Eq. 2). If the Ratio of Number of Stations (RNS) equals 1, then IDs and departures of every station pair at the same position of the station sequence are comparable. Else, all metrics are set 0.

$$\text{Ratio of Number of Stations (RNS)} = \frac{\text{Number of stations in trip 1}}{\text{Number of stations in trip 2}} \quad (2)$$

$$\text{Ratio of Identical Stations (RIS)} = \frac{\text{Number of identical stations at the same position in trip 1 and 2}}{\text{Number of stations in trip 1 or trip 2}} \quad (3)$$

$$\text{Ratio of Identical Departures (RID)} = \frac{\text{Number of identical departures at the same position in trip 1 and 2}}{\text{Number of stations in trip 1 or trip 2}} \quad (4)$$

These metrics can be used to make an initial case distinction between pairs of trips in order to filter the relevant cases (i.e. differences between the methods) and analyse them in detail. Since only combinations of RIS = 1 and RID = 1 occur in the available data (i.e. if the station sequence is the same, the departure times are also the same), these can be combined into one characteristic. Accordingly, only the characteristics Route (sequence of edges) and Stops (sequence of stations and departures), which are either equal (metric value = 1) or different (metric value < 1), will be used to differentiate cases in the following:

- Case 1: Route equal & Stops equal
- Case 2: Route different & Stops equal
- Case 3: Route equal & Stops different
- Case 4: Route different & Stops different

The result of the case distinction is given in **Table 5** for all bus lines and in **Table 6** for all tram lines. Since case 3 (Route equal & Stops different) never occurs, the corresponding column is not listed. The last column indicates the ratio of completely identical trips (case 1) to the total (comparable trips) and is an indicator of how often both methods reproduce the same route. Whether this route is also correct has not yet been systematically checked and proven, as data on the basic truth is missing or not available in the required form. Further studies to validate our new method are therefore to follow. Finally, selected examples from the random visual inspection of the routes are illustrated and assessed with regard to correctness (see **Figure 10**, **Figure 11**, **Figure 12** and **Figure 13**).

***Table 5**. Number and ratio of identical bus trips, by comparing trips from GTFS2PT and our new method in terms of equal edge sequence (Route: R) and equal sequence of stops and departures (S)*

	Bus Line (MVB only)	Number of Trips planned	Identical Trip IDs (comparable)	R equal S equal (case 1)	R different S equal (case 2)	R different S different (case 4)	Identical Trips Ratio [%]
1	51	95	95	87	0	8	91.6
2	52	106	106	0	52	54	0.0
3	53	95	95	48	47	0	50.5
4	54	115	115	25	46	44	21.7
5	56	42	40	22	18	0	55.0
6	57	102	102	0	72	30	0.0
7	58	110	110	51	5	54	46.4
8	59	11	11	0	5	6	0.0
9	61	45	45	4	3	38	8.9
10	66	37	37	18	18	1	48.6
11	69	122	122	0	122	0	0.0
12	71	146	146	40	70	36	27.4
13	72	110	110	32	31	47	29.1
14	73	127	127	13	114	0	10.2
15	N3	16	0	0	0	0	0.0
16	N4	16	8	0	0	8	0.0
17	N5	17	0	0	0	0	0.0
18	N6	16	8	0	0	8	0.0
19	N7	18	10	0	10	0	0.0
20	N9	14	14	6	5	3	42.9
	Total	1360	1301	346	618	337	26.6

***Table 6**. Number and ratio of identical tram trips, by comparing trips from GTFS2PT and our new method in terms of equal edge sequence (Route: R) and equal sequence of stops and departures (S)*

	Tram Line (MVB only)	Number of Trips planned	Identical Trip IDs (comparable)	R equal S equal (case 1)	R different S equal (case 2)	R different S different (case 4)	Identical Trips Ratio [%]
1	1	204	0	0	0	0	0.0
2	2	190	190	0	183	7	0.0
3	4	201	190	0	0	190	0.0
4	5	120	113	0	0	113	0.0
5	6	198	191	0	185	6	0.0
6	9	204	198	0	0	198	0.0
7	10	192	93	0	0	93	0.0
8	N1	16	0	0	0	0	0.0
9	N2	16	16	0	8	8	0.0
10	N8	16	16	0	16	0	0.0
	Total	1357	1007	0	392	615	0.0

The first example illustrates case 1 with two different bus lines on the left and right of **Figure 10**. The green line indicates the route calculated by our new method, whereas the route from GTFS2PT is red and not visible in this case, as it is totally overlayed by the green line. Consequently, there are no differences in terms of the route. The stops are not explicitly illustrated here but also identical.

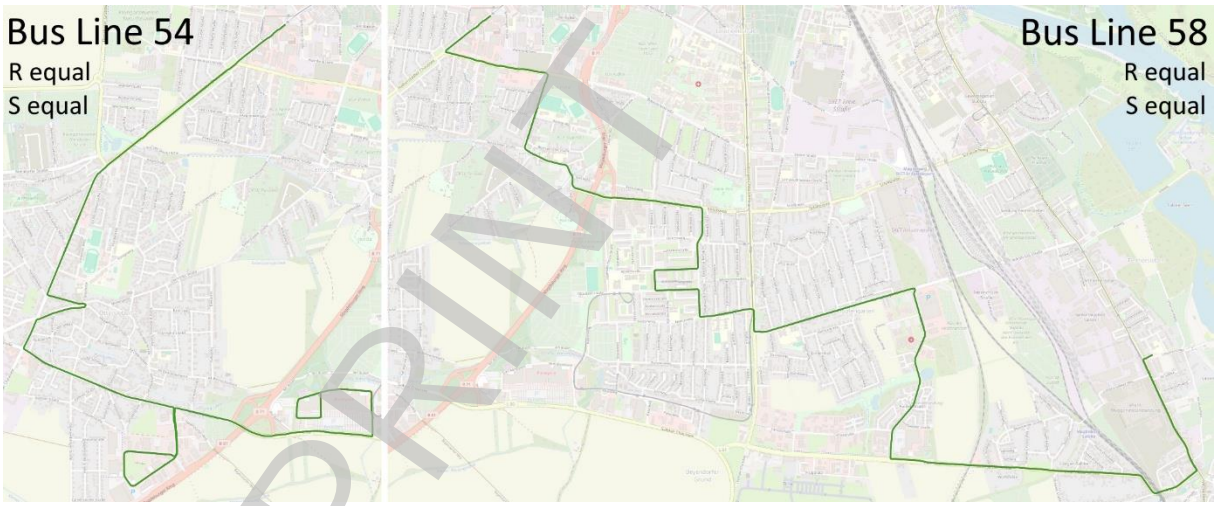

***Figure 10.** Examples of case 1 (Route equal, Stops equal) with a trip of bus line 54 (left) and 58 (right)*

The second example illustrates case 2 with a bus line, whose total route from south to north is depicted on the left of **Figure 11**. The interesting section of differences is framed and enlarged on the right. The red coloured section of the GTFS2PT route deviates from the correct route (green line) in this area, although the sequence of stations is identical. However, the positions of two stations in the selected area (one in the north and one in the south) are not correct, after they were relocated by GTFS2PT (according to algorithm 1, as explained in Section 3.1) to the specified points S. Although there are further stops on the red line, which are illustrated as red rectangles, they are not served by the bus line, that is rather atypical and should also be regard as indication of an incorrect route. As this incorrect route section is noticeably longer and departure times remain unchanged, delays at the second, northern stop and after that are more likely to happen in the simulation runs.

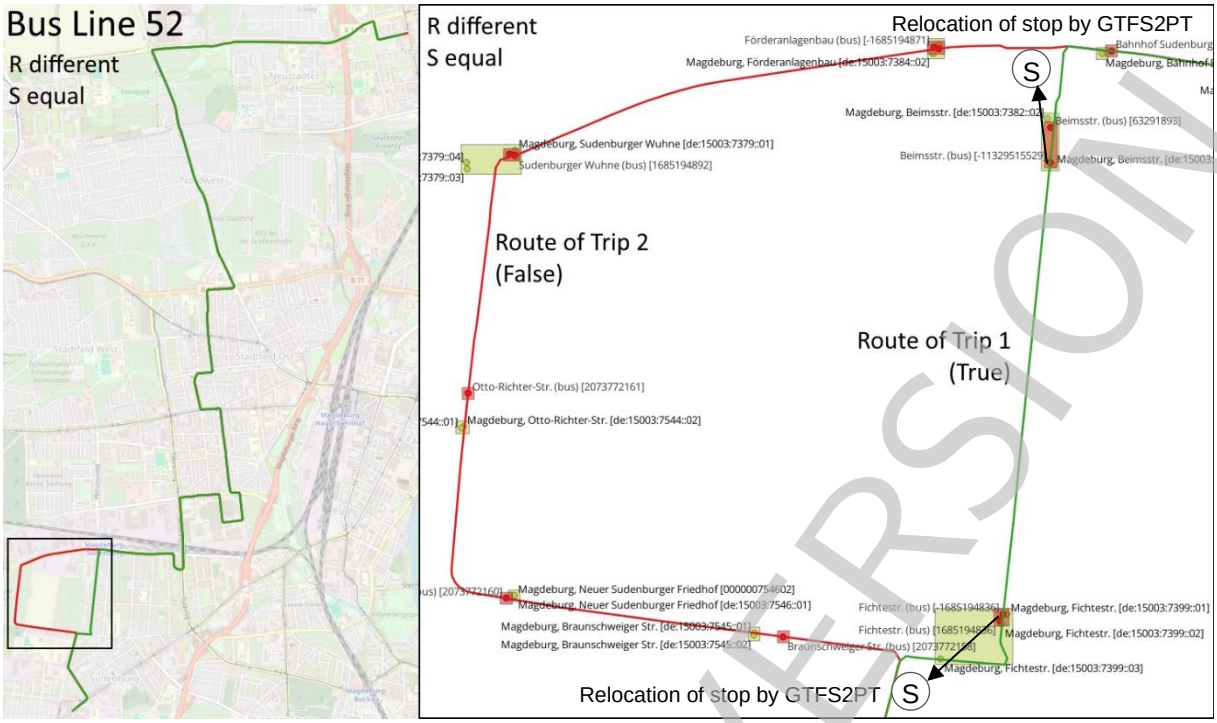

Figure 11. *Example of case 2 (Route different, Stops equal) with a trip of bus line 52*

The third example is illustrated in **Figure 12**, which is also divided into an overwiew of the total route on the left and a detailled map of the area of route differences on the right. Here, the incorrect route section of GTFS2PT (red line) is shorter than the correct route section (green line), which is also calculated by our new method. Moreover, the station sequences also differ in this area, as three stations are not served (each depicted as bundle of rectangles on the green line), so that this example represents case 4. In contrast to case 2 and the previous example, GTFS2PT did not relocate the respective stops or used the upper station on the incorrect route as substitute. Since departure times at the remaining stations before and after the incorrect route section are retained, there is a correspondingly larger time gap (here of 6 minutes) between the respective stations, so that earlier arrivals and longer stop durations are more likely to happen in the simulation runs.

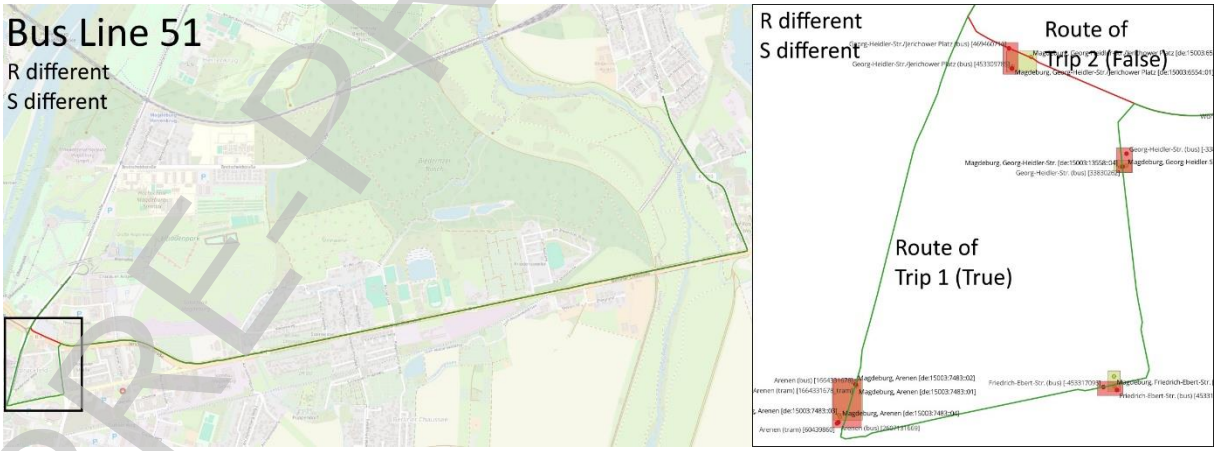

Figure 12. *Example of case 4 (Route different, Stops different) with a trip of bus line 51*

At first glance, trams do not have any full matches of routes and station sequences calculated by GTFS2PT and our new method, respectively (cf. case 1 in **Table 6**). When we look deeper into the routes and stations of comparable trips, the differences usually result from a

slight repositioning of stops at the end of a route and a corresponding route shortening by GTFS2PT, as illustrated in **Figure 13** using the example of tram line 2. The same situation at termini also occurs in some cases with bus lines (e.g. at the upper end of the route of bus line 52 in **Figure 11** on the left). If this slight difference at the termini is always neglected, all trips of tram line 2 and 6, which are currently assigned to case 2, will be identical (case 1). However, since the repositioning of stops is not mandatory in these cases and should rather be avoided by GTFS2PT (i.e. algorithm 1), the classification into case 2 is retained for the moment.

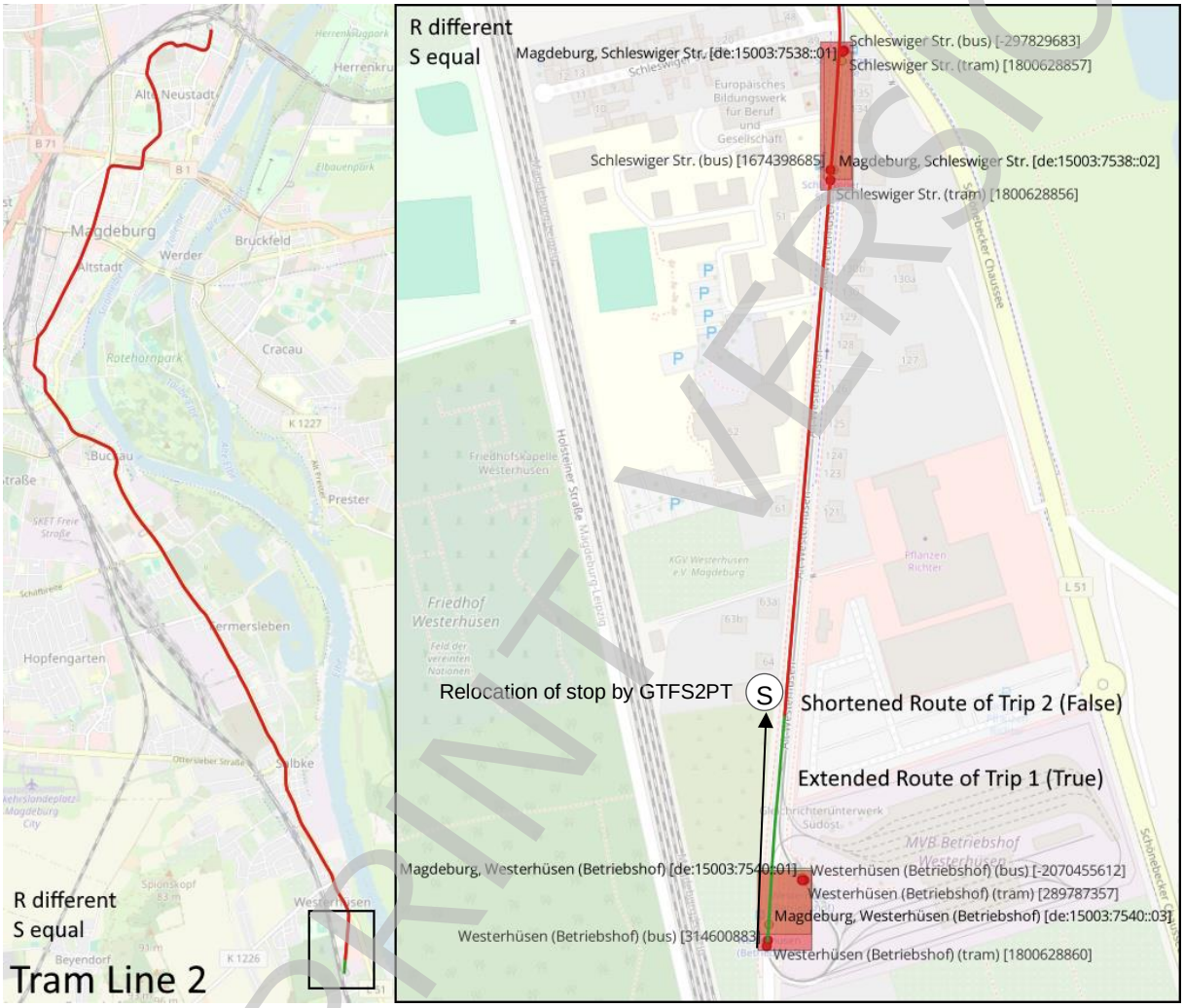

Figure 13. *Example of case 2 (Route different, Stops equal) with a trip of tram line 2*

Other differences in tram routes result from completely missing route sections, that are not applied by GTFS2PT, such as the short intermediate connection at position 2 in **Figure 9** d for morning and evening depot trips of line 2 (case 4), or the connection to the upper terminus (position 1) for all trips of line 1. The latter are therefore not mapped by GTFS2PT at all.

3.4 Computing Time

Although our new method has been already implemented in Python to retrieve the result data presented above, when we were writing the scripts we rather focused on high accuracy and correctness of results than high performance of the programme by particularly minimising the computing time. Therefore, the performance is object to future work, since optimisation and implementation of our new method is an iterative and on-going process. Nevertheless, we measured the current computing time of running all programme parts (i.e. different Python scripts), which are assigned to respective phases of the method, as depicted in **Table 7**.

***Table 7**. Computing time of our new method*

	Phase	Main Purpose	Frequency of Use (expected due to external updates)	Computing Time [seconds]
1	GTFS Data Processing	Retrieval and preparation of timetables etc.	weekly or less frequent (depends on GTFS data provider)	270
2	OSM Data Processing	Retrieval and preparation of stops routes (map data), Download and generation of SUMO model	one-time / if required (depends on changes in stops and routes of bus and tram lines in the study area)	593
3	GTFS and OSM Data Fusion	Clustering and matching of Stops, Routing of lines/trips	if phase 1 or 2 updated	89
4	Writing Route File for SUMO	Preparation of simulation in sUMO	if phase 3 updated	26
5	Writing Shape Files for QGIS	Preparation of routes verification in QGIS	one-time / if required	60
	Total			1038 s (17.3 min)
All measurements were repeated 5 times and performed by a Lenovo ThinkStation P3 (i7).				

The total computing time amounts to approximately 17.3 minutes, that is about 35 times higher than the respective computing time of GTFS2PT (approx. 30 seconds). However, our new method also includes download and generation of the SUMO model itself (phase 2), that is often time-consuming (up to the size of study area, here 7.3 minutes) and not performed by GTFS2PT. When excluding this task and all other tasks from phase 2, which are usually executed once only (if OSM data of the study area remain unchanged), the total computation time decreases to 445 seconds (7.4 min). Nevertheless, this value is obviously still too high with respect to user acceptance, and therefore needs to be reduced in future by optimisation of the Python programme.

3.5 Summary and Interpretation of Findings

The comparative assessment on aggregated and detailed levels reveals that our new method is able to calculate accurate routes of all tram trips (100%) and almost all bus trips (99.3%) (cf. **Table 3** and **Table 2**, respectively). In contrast, GTFS2PT particularly failed in tram trips from or to depots with partly different routes, which usually deviate from the regular route. These trips are either not included or have an incorrect route in the respective route file of GTFS2PT. This is assumed to be the reason, why some trips are always missing in 4 of 10 tram lines (cf. **Table 3**, right column, row 3-6). Although all trips of line 2 (row 2) are included in the route file of GTFS2PT, a closer look into the network usage reveals, that these trips have an incorrect route, since the correct route section is never used (cf. **Figure 9** d, position 2). Since such depot trips are not applicable to buses in the city of Magdeburg, the error only occurs at trams. The detailed analysis reveals more differences between both methods and detects errors which are not obvious on the aggregated level, for example at bus line 52, whose route is always partly incorrect (cf. **Table 2**, row 2, **Table 5**, row 2, and **Figure 11**). Overall, our new method is obviously more accurate than GTFS2PT in case of trams, but also in case of buses. Remaining errors at two bus lines, which applies to 10 of 1360 trips (cf. **Table 2**, row 5 and 19), are always made transparent as they are not included in the route file. The suspected reason for this error has been already identified (i.e. some required bus stops are missing in OSM data) and will be fixed by further improving the method in future.

4 Conclusion and Outlook

In this article, we design a multi-part algorithm for processing timetable data from public data sources. The processed data is used in the traffic simulation model SUMO. We get each route for each individual trip of public transport vehicle. This is necessary because occasionally deviating routes of bus and tram lines, such as trips in the morning and evening to depots or temporary diversions are only shown in the GTFS timetable data, but not in OSM route data.

With our new method we were able to determine all tram trips that took place on a selected day, whereas with GTFS2PT we did not get all trips for individual lines (cf. **Table 3**). Among the trips determined, however, a closer look at the routes revealed clear differences between the methods and in some cases also verifiable errors of GTFS2PT. For example, depot trips of tram line 2 were indicated in the route file, although the special section of route required for this trip is not included (cf. position 2 in **Figure 9** d). In total, 346 of 1357 tram trips from the route file were missing with GTFS2PT, while all of them could be constructed with out method.

For bus trips, the numerical differences between the two methods were much smaller (cf. **Table 2**). However, a closer look at the routes of individual bus trips revealed differences, some of which point to errors by GTFS2PT. For example, individual stops are moved to locations that do not exist in reality (cf. **Figure 11** and **Figure 13**), because the underlying algorithm 1 of GTFS2PT considers stop location data from GTFS only. As we pointed out in [6], stop location data from OSM is often more accurate than from GTFS (cf. Section 1).

Regarding timetable data we assume that the GTFS data is always correct. However, for example, a night bus line is incorrectly shown as a tram line in the city of Magdeburg. Thus, since two stops of the line are not on the tram network, the error is automatically detected. However, since several solution options may exist for this error, an automatic correction is not possible so far. For instance, one option is correcting the location of stops (if this changes the route only slightly), and another option is correcting the vehicle class (i.e. bus instead of tram).

These examples and all previous experiences indicate that a graphical representation of the routes and stations on a map (e.g. in the open source software QGIS) is recommended for manual checking and correction of errors by each SUMO user. Although errors can be automatically detected by comparing data from different sources (GTFS versus OSM) with plausibility checks, automatic correction is much more difficult to implement, especially if there are several solution options. This would only be conceivable with the help of statistical models (AI), which, however, are rather unreliable and not transparent compared to deterministic methods. From a realistic and pragmatic point of view, the option of automatic map display seems feasible and useful in the foreseeable future as supporting tool for the user to manually correct erroneous routes. In this respect, our new method will be further developed and improved in this direction.

Data availability statement

The main input data of the SUMO model and our new method, which is presented in this article, is publicly available from OpenStreetMap [10] and Open Data ÖPNV [12].

Funding

Research covered in this article benefits from participation in the project IMIQ (Intelligent Mobility Space in the District) which is an interdisciplinary research and development project of the IMR initiative and located at the Science Port in Magdeburg (from 01/2024 to 12/2027). The IMIQ project is funded by EFRE – European Regional Development Fund.

Author contributions

Alexander Kaiser: Conceptualization, Methodology, Software, Data Curation, Formal Analysis, Validation, Visualization, Writing – Original Draft, Writing – Review & Editing.
Alexander Schmaus: Writing – Original Draft, Writing – Review & Editing.

Competing interests

The authors declare that they have no competing interests.

References

[1] M. Schweppenhäuser, T. Großmann, K. Schrab, R. Protzmann, and I. Radusch, "Modeling Bus Traffic for the Berlin SUMO Traffic Scenario," *SUMO Conference Proceedings*, vol. 6, pp. 101–115, 2025, doi: https://doi.org/10.52825/scp.v6i.2613.

[2] Otto-von-Guericke-Universität Magdeburg. "IMIQ - Intelligenter Mobilitätsraum im Quartier." [Online]. Available: https://www.niimo.ovgu.de/IMIQ.html

[3] P. A. Lopez et al., "Microscopic Traffic Simulation using SUMO," in *The 21st IEEE International Conference on Intelligent Transportation Systems,* IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/.

[4] A. Schaffland, J. Nelson, and J. Schöning, "Simulating Traffic Networks: Driving SUMO towards digital twins," *SUMO Conference Proceedings*, vol. 5, pp. 113-125, 2024, doi: 10.52825/scp.v5i.1105.

[5] M. Pechinger and J. Lindner, "Sumonity: Bridging SUMO and Unity for Enhanced Traffic Simulation Experiences," *SUMO Conference Proceedings*, vol. 5, pp. 163–177, 2024, doi: 10.52825/scp.v5i.1115.

[6] A. Kaiser and A. Schmaus, "Processing of Public Transport Data from GTFS and OSM for an Up-to-Date Simulation of Trips in SUMO," in *Transport Systems Development – Methods and Solutions*. Cham: Springer, 2026, pp. 25–46, doi: 10.1007/978-3-032-14826-1_3.

[7] H. Blache and N. Saunier, "Is It Possible to Automatically Build a Large Scale Metropolitan Traffic Model? Evidence from a Study of Connected Transportation Applications in Montreal" in *Transportation Research Procedia*, vol. 82, pp. 495-514, 2025, doi: 10.1016/j.trpro.2024.12.057.

[8] L. Codeca, R. Frank, and T. Engel, "Luxembourg SUMO Traffic (LuST) Scenario: 24 Hours of Mobility for Vehicular Networking Research," *in 2015 IEEE Vehicular Networking Conference (VNC)*, 2015, pp. 1–8, doi: 10.1109/VNC.2015.7385539.

[9] K. Schrab, R. Protzmann, and I. Radusch, "A Large-Scale Traffic Scenario of Berlin for Evaluating Smart Mobility Applications," in *Smart Energy for Smart Transport*. Cham: Springer, 2023, pp. 276–287, doi: 10.1007/978-3-031-23721-8_24.

[10] OpenStreetMap contributors. "Planet dump retrieved from https://planet.osm.org." 2017. [Online]. Available: https://www.openstreetmap.org/.

[11] MobiltyData. "General Transit Feed Specification (GTFS)." [Online]. Available: https://gtfs.org/

[12] OpenData ÖPNV and Verkehrsverbund Rhein-Ruhr. „Deutschlandweite Sollfahrplandaten (GTFS)." [Online]. Available: https://www.opendata-oepnv.de (accessed: 26.01.2026)

[13] G. Cich, et al., "Data Preparation to Simulate Public Transport in Micro-Simulations Using OSM and GTFS," in *Procedia Computer Science*, vol. 83, pp. 50-57, 2016, doi: 10.1016/j.procs.2016.04.098.

[14] J. Vuurstaek, et al., "GTFS Bus Stop Mapping to the OSM Network," in *Procedia Computer Science*, vol. 109, pp. 50-58, 2017, doi: 10.1016/j.procs.2017.05.294.

[15] German Aerospace Center (DLR) and others. „ SUMO User Documentation - GTFS - gtfs2pt.py." [Online]. Available: https://sumo.dlr.de/docs/Tools/Import/GTFS.html (accessed: 10.02.2026)