STRASSE UND VERKEHR NR. 7-8, JULI/AUGUST 2017
ROUTE ET TRAFIC N° 7-8, JUILLET/AOÛT 2017

Public Transit Mapping on Multi-Modal Networks in MATSim

Public transit vehicles, such as buses, interact with private traffic. They can be stuck in traffic which leads to delays or they can cause traffic jams if they stop at on-street bus stops. To see such effects in a transport simulation, the route a bus takes on a network must be known. However, public transit schedule formats usually do not provide information on which roads a bus actually uses. Normally, only stops and their sequences are available. This thesis proposes an algorithm to automatically generate the route of public transit vehicles.

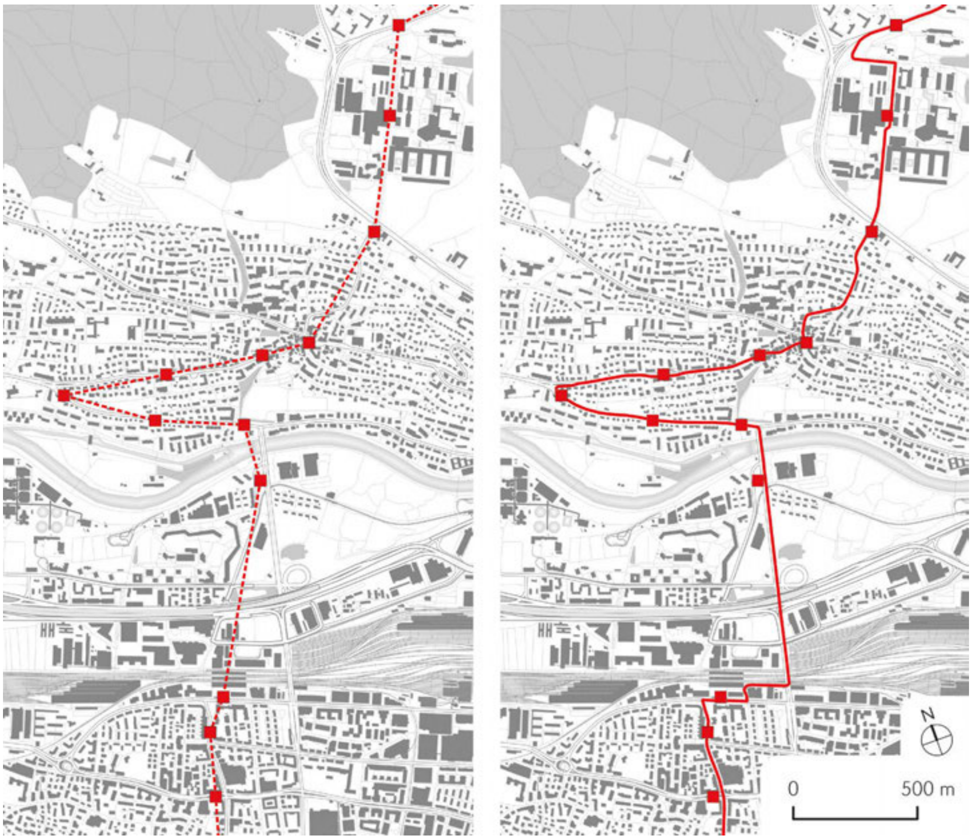
VON
FLAVIO POLETTI
Verkehrsingenieur,
TEAMverkehr.zug ag, Cham

VSS-Preisträger 2017

Flavio Poletti wurde für diese Masterarbeit an der ETH Zürich mit dem VSS-Preis 2017 ausgezeichnet. Er ist einer von drei Preisträgern.

In Switzerland, public transit schedules are available in two data formats: HAFAS and GTFS. While HAFAS is a popular format in Central Europe, most of publicly available schedules worldwide are provided in GTFS. Both formats offer stop sequences with departure and arrival times for transit routes of different modes (bus, rail, ferry, funicular, etc.). However, just the order of the stops and timestamps is not enough to simulate the interactions between busses and cars. For each bus, the exact route on the network and the links the vehicle stops on must be known. While GTFS allows to store network routes of vehicles as polylines, most feeds do not provide this data at this stage. Therefore, network routes must be gathered or created separately. This thesis focuses on reproducing existing bus routes in a simulation and not on optimizing or otherwise improving routes.

Several approaches to create network routes for public transit vehicles come to mind. First, transit agencies could be contacted directly and asked for data about network routes. However, it is not given that agencies have more data in the first place and even then, collecting possibly different data formats from multiple agencies can be cumbersome. Second, one could create network routes manually based on maps or knowledge of the area and the transit lines. Manually digitizing and assigning network



1 | Example for the input stop sequence of a transit route (bus line 80 in Zurich) on the left. The image on the right shows the route created by the proposed algorithm (map: OpenStreetMap).

routes to transit vehicles is very time-consuming. Thus, this approach is limited to small areas as it becomes very expensive for large schedules. A third approach is to use GPS data to create network routes. Extensive literature on mapping GPS data to a network is available. However, gathering GPS track data, even for small areas with a limited number of vehicles, is expensive and simply not viable for large scenarios like the whole Swiss public transit schedule. Conversely, conventional GPS mapping approaches, using only the stop locations as «GPS data points» without further data points between stops, do not work because point density is too low to achieve acceptable network routes. In addition, coordinates of stop locations are often generalised and do not represent the location where vehicles stop or passengers board. In many schedules, multiple stop locations on different roads or for different directions are combined to one parent stop with the same name. This often means that the closest road segment to a stop is not the correct one for all transit routes passing a stop.

Therefore, the approach pursued in this thesis is generating network routes directly from the stop sequences using an algorithm with no additional input. Such an algorithm has been developed and implemented for MATSim, a framework for multi-agent transport simulations which is co-developed at ETH Zurich.

Network data source

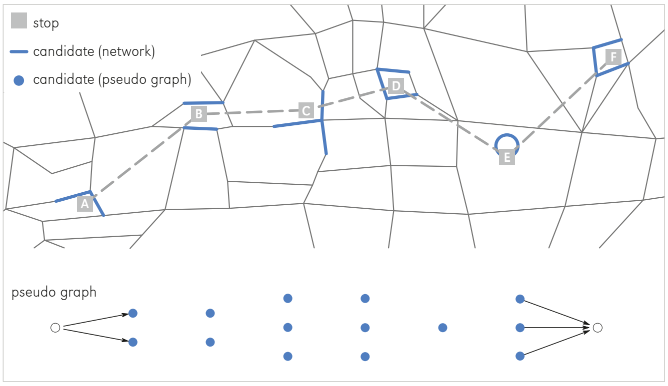
To map a public transit schedule to a network, the network needs to be created as well. Networks for MATSim are normally generated from OpenStreetMap (OSM) which is a free and editable map of the world, released with an open content license. OSM has proven to be a valuable data source for network creation for different simulation scenarios worldwide. The often proprietary nature of other sources for network creation (e.g. swisstopo's Vector25) makes it difficult to share the generated networks. In addition, it is usually necessary to combine multiple data sets to reach the level of information OSM offers in one source.

OSM can not only be used as a source for network creation but also for public transit data. OSM offers tags to define spatial public transit data such as routes and stops. However, time information (i.e. departure times for each stop) cannot be tagged. Using OSM data in a fully automated process to create public transit schedules is not yet feasible. Data is very inconsistent as there is currently no consensus within the OSM community how to map and tag public transit data. The level of detail varies between regions. While stop locations are normally accurate, often parts of a route are missing or misplaced which makes automatic processing of route information difficult.

Algorithm

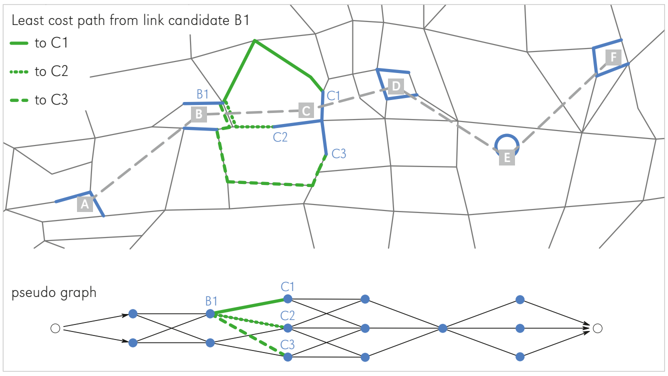
The proposed algorithm ("pseudo routing algorithm") requires only minimal input. It requires a schedule in which: first, each transit route has a sequence of stops and second, each stop has coordinates. The algorithm calculates the least-cost path from the transit route's first to its last stop with the constraint that the path must contain a link candidate for every stop of the stop sequence. For each transit route, the algorithm consists of the following steps:

1. Identify possible link candidates for each stop. Link candidates are road segments close to the stop. If no links are nearby (Stop E in Fig. 2), an artificial loop link is created.
2. Create a pseudo graph using the link candidates as nodes. Add a dummy source and destination node to the pseudo graph.



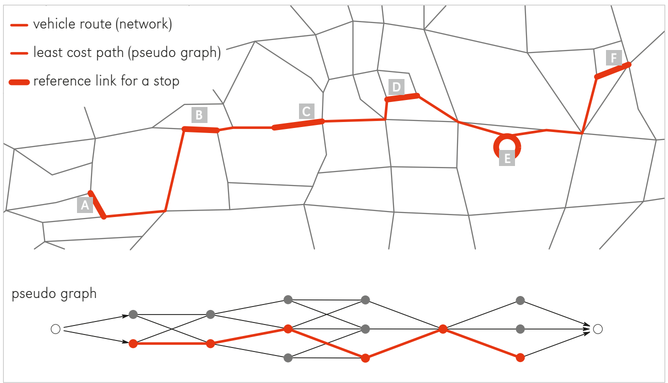
2 | Step 1 and 2 of the pseudo routing algorithm.

3. Calculate the least-cost path between each link candidate pair. This path is represented by an edge in the pseudo graph, connecting two link candidate nodes. The edge weight is the path's travel cost plus half the travel cost of the two link candidates it connects.



3 | Step 3 of the algorithm.

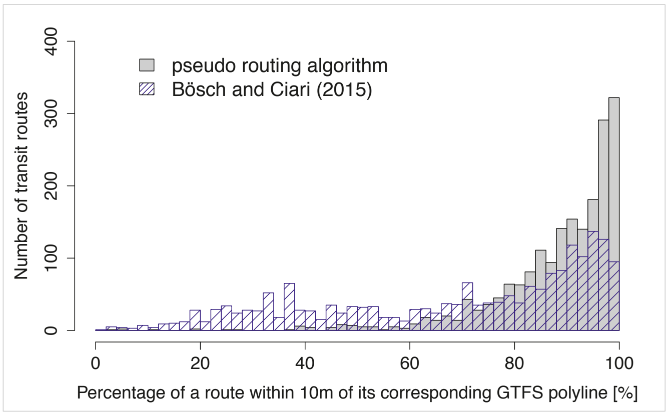
4. Calculate the pseudo least-cost path from the source node to the destination node in the pseudo graph. The resulting least-cost path contains the best fit link candidate for each stop.
5. Create the vehicle route on the network. Each stop is referenced to a link, which is given by the link candidate that is part of the pseudo least-cost path. The least-cost path on the real network between the referenced links is used to create the network path for the transit route.



4 | Step 4 and 5 of the algorithm.

Analysis

- The results of the implemented workflow have been validated by testing the accuracy of the mapping.
- The tests have been conducted with schedules based on a GTFS feed from 2016 for the Zurich area. The feed is provided by the Zürcher Verkehrsverbund (ZVV) and covers all bus, tram, funicular and ferry routes of the Zurich area.
- The feed contains the shapes of the trips for bus routes, i.e. a polyline representing the vehicle's path. These polylines were used to validate the schedule created by the pseudo routing algorithm. Note that it has been assumed that these polylines are correct. The feed has been converted to an unmapped MATSim transit schedule (i.e. a schedule without network routes). This unmapped schedule has then been mapped to a network (created from OSM) using the pseudo routing algorithm. The algorithm's result was also compared to another mapping algorithm for MATSim by Bösch and Ciari (2015).



5 | More than half of all transit routes (1087 of 1926) lie within a 10m buffer of the corresponding polyline for at least 90 % of their length.

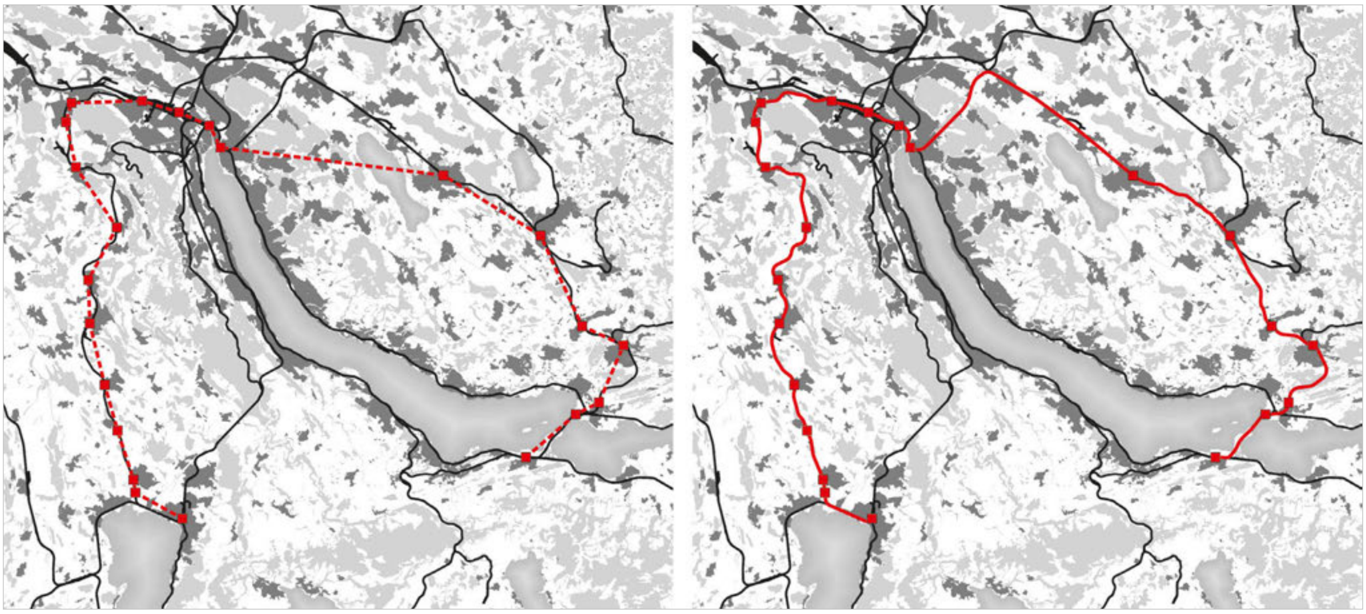
Results

The mapping accuracy tests have shown that, overall, the vast majority of routes were mapped very similarly to their corresponding GTFS polylines. The accuracy has also been improved compared to previous implementations.
Mapping results were not satisfying for routes whose stops are farther apart, like express lines that skip some stops. If the network situation around a stop is complex (e.g. high number of links within a small radius or links on multiple street lev-

DE

Kartierung des öffentlichen Personennahverkehrs in multimodalen Netzen in MATSim

Fahrzeuge des öffentlichen Personennahverkehrs, wie beispielsweise Busse, interagieren mit dem privaten Verkehr. Sie können in einem Stau stecken, der zu Verspätungen führt, oder selbst Staus verursachen, wenn sie an direkt auf Strassen befindlichen Haltestellen halten. Um solche Auswirkungen in einer Verkehrssimulation zu sehen, muss die Route bekannt sein, die ein Bus in einem Netz fährt. Jedoch ist aus den Busfahrplänen in der Regel nicht ersichtlich, welche Strassen ein Bus tatsächlich benutzt. Normalerweise geben die Fahrpläne nur Haltestellen und ihre Reihenfolge an. Diese Abhandlung präsentiert einen Algorithmus zur automatischen Generierung der Fahrstrecke von Fahrzeugen des öffentlichen Personennahverkehrs.



6 | While the algorithm focuses on bus routes, it can also be applied to rail routes, for example the line S5 between Zug and Pfäffikon SZ (map: OpenStreetMap).

els), it is likely that not enough link candidates are selected. This leads to incorrect paths because the right link candidates were not part of the pseudo graph. In the implemented algorithm a link cannot be a candidate for two subsequent stops. This might lead to unexpected results if there are too many stop facilities along a single link. It should also be noted that the mapping quality largely depends on a consistent and accurate network. If links are missing, especially bus lanes, the result is likely to be wrong.

Most problems with routes (loops or simply wrong routes) come from a wrong selection of link candidates. The implemented approach takes a fixed number of links within a given radius. More complex link candidate search functions are conceivable, for example depending on the number of transit routes on a stop or on the type of stop. One could improve link candidate selection by including OSM data to order link candidates. Links that are close to a stop location identified in OSM could get a higher score. This does not even require matching data sets.

A second improvement step would be to develop more complex least cost path algorithms for the network. This would improve mapping without changing the basic algorithm. The implementation uses a Dijkstra-based algorithm which allows two types of link travel costs: link length and travel time. Additional data could be included to calculate the travel cost. If GPS data is available, links with GPS points close to them could have decreased travel costs. Again, OSM data could also be included. For example, buses could have lower travel costs if they travel on links that are tagged as bus routes in OSM. In addition, the implemented least cost path algorithm allows U-turns which should lead to a travel cost increase in further development steps.

OSM in general could provide much more information on networks even beyond public transit. For example, pedestrian and level crossings or traffic signal locations could be used to adapt link capacities or free speed values.

Conclusion

The test results show that the implemented algorithm is a viable way to automatically generate network routes for public transit schedules. Large data sets with several thousand transit routes can be processed fast. The mapping results provide an adequate basis for transport simulations. However, it is still an automatic algorithm so the resulting network routes need to be checked for plausibility. Fixing incorrect mappings is still less costly than creating all routes by hand.

FR

Cartographie des transports en commun locaux dans des réseaux multimodaux avec l'application MATSim

Les véhicules des transports en commun locaux, comme par exemple les bus, interagissent avec le trafic privé. Ils peuvent se retrouver dans les embouteillages générant des retards ou même provoquer eux-mêmes des embouteillages lorsqu'ils s'arrêtent à des arrêts se trouvant directement sur la chaussée. Pour évaluer de telles conséquences dans une simulation de trafic, il faut connaître l'itinéraire du bus dans le réseau. Mais en règle générale les horaires de bus ne montrent pas les routes empruntées réellement par un bus. Normalement, les horaires de bus n'indiquent que les arrêts et leur chronologie. Ce document présente un algorithme de génération automatique du trajet de véhicules des transports en commun locaux.