SEMI-AUTOMATIC TOOL FOR MAP-MATCHING BUS ROUTES ON HIGH-RESOLUTION NAVIGATION NETWORKS

S. ORDONEZ \textsuperscript{a} and A. ERATH \textsuperscript{b}

\textsuperscript{a} Future Cities Laboratory (FCL), Singapore-ETH Centre for Global Environmental Sustainability (SEC), Singapore
Email: ordonez@ivt.baug.ethz.ch

\textsuperscript{b} Future Cities Laboratory (FCL), Singapore-ETH Centre for Global Environmental Sustainability (SEC), Singapore
Email: alexander.erath@ivt.baug.ethz.ch

ABSTRACT

A map-matching approach is presented that combines public bus routes information (sequences of consecutive stop locations and sequences of geo-referenced points) with a high resolution network. It is especially designed to account for a) low temporal resolution of geo-referenced points and b) to allow that a stop which is served by several lines is attached to one single link only. Taking into account fixed link-stop relationships and other parameters an automatic solution using the A* search algorithm between consecutive stops is generated for each route. The link cost takes not only the travel time into account but also the distance to the geo-referenced points. For final corrections, the approach includes a user interface which allows visualizing and modifying the network and the sequences interactively. It is possible to add or remove links, replace link-stop relationships, and review the correctness of the result. The effectiveness of the approach is reported.

Keywords: Map-matching, Public bus system, High-resolution navigation network, Semi-automatic

INTRODUCTION

Current public transport assignment models adapt network assignment models to work for public transport traffic. Many commercial software products like EMME, VISUM and OmniTRANS offer sophisticated procedures that include timetable-based route search. However, these models do not include the interaction between public transport services and private transport. The multi-agent, activity-based transport simulation model MATSim has recently been expanded to include public transport (Rieser, 2010). The implementation handles private car traffic and public transport traffic in an integrated way and is applicable to large-scale scenarios with millions of agents. This allows accounting for dependencies between the different modes which are of special importance in urban regions where buses and tramways often have to share lanes with private car traffic. A central requirement for such a model is the accurate routing of public transport lines on the transport network. Whereas this is usually straightforward for rail based public transport modes, the routing problem for buses requires more attention as experience shows that the assumption of a shortest-path between two consecutive stops leads to unsatisfactory results. To overcome this shortcoming, one can either rely on expert knowledge and facilitate the routing manually or employ map-matching algorithms that depend on tracking data. Since the first is rather burdensome and due to the increasing availability of GPS tracking data counterintuitive, map-matching is becoming more and more relevant. However, common map matching algorithms are usually not designed to account for peculiarities of public transport routing. Besides, such procedures are very sensitive to errors in network coding, inaccuracies in bus stop locations, and simplified link shapes in the network model.

This paper presents a semi-automatic procedure that combines public bus routes information (sequences of consecutive stop locations and sequences of geo-referenced points) with a high resolution network for the case of Singapore. The objective is to obtain a sequence of links for every
route of the 355 bus lines, and to relate each of the 4584 bus stops with one single link in the network. The paper is structured as follows: First the specific problem is defined explaining the special features of the case. Then, the semi-automatic solution approach is introduced, followed by the description of the automatic algorithm, the automatic verification process, manual editing functionalities and the developed software. Afterwards, the results when applying the procedure to the Singapore case are shown presenting finally an outlook and conclusions from the experience. Thus, the effectiveness of the approach is reported and strands for further improvement are evaluated.

1. RELATED WORK

This section is based on Schüssler (2010). To obtain paths from GPS point sequences, several automatic procedures have been developed in recent years. A first approach is based only in geometric information: position of GPS points vis-a-vis to network elements. The simplest strategy consists of searching the nearest node or link using the Euclidean distance. Taking into account the GPS points sequence, it is possible to define curves between each pair of consecutive points. Thus, the distance and/or angle between this curves and the links can be minimized, as it was proposed by White et al. (2000). The problem with geometrical approaches is that GPS time and network topology information is ignored. These procedures are very sensitive to outliers and depend a lot on a good match between heterogeneous data sources.

A second manner for dealing with this problem is based on topological procedures. Most of these use geometric approaches in an initial step for finding the first link of the path, performing afterwards a selection process between the succeding links of that link. For the choice of the next link the most common criterion is the perpendicular distance between some next GPS point and the candidate link (point-segment distance). Other selection criteria are the difference between the angle of a segment defined by some two consecutive GPS points and the candidate link angle (Greenfeld, 2002), or the angle between the candidate link and the line that connects the start node of this link and some of the following GPS points (Quddus et al., 2003). If several criteria are used usually each one is weighted and a function is calibrated (Velaga et al., 2009). After the selection step, the whole procedure is performed again for determining the next link of the sequence. Special treatment of outliers (Greenfeld, 2002), null selections or gaps (Chung and Shalaby, 2005; Tsui and Shalaby, 2006), and other post-processing routines can be applied. These methods have lower computation times as working link sets are limited and are less sensitive to outliers. However, depending on the network topology a bad link selection may derive a complete wrong route result from that point onwards.

Since then, more advanced strategies have been developed. Some of them are based on the idea of taking into account confidence regions around GPS points (Velaga et al., 2009). Other one proposed by Nielsen et al. (2004) uses the Dijkstra’s algorithm for finding the shortest path as a sequence of nodes. Starting from a calculated first node the cost of each subsequent node is the perpendicular distance from the GPS points between the current node and the subsequent one to the link defined by these two nodes. The problem is how to determine which are the GPS points between each pair of nodes. Furthermore, all previous algorithms have the problem about the necessity of confidence in the previous selection for continuing. This results in strategies based on using memory for saving several scored solutions through the process, first with Pyo et al. (2001) who use the Multiple Hypothesis Technique and continuing with Marchal et al. (2005) who control the size of the candidates solution set with a fixed limit. Schüssler (2010) refined Marchal’s algorithm by splitting the GPS points sequence into several subsequences previously. In a given subsequence, if two consecutive GPS points are relatively far the subsequence is divided at this point and so on. Marchal’s algorithm is then applied to each subsequence and in the end a gaps treatment procedure is performed. She also improved the identification of the initial link.

Advanced procedures present acceptable results when a high temporal or spatial resolution of GPS points according to the network link size area available (at least 10 points per link). As the public
transport information not always present this quality, this work takes a simpler automatic approach taking some of the ideas summarized previously, but including editing functions for solving special cases. Furthermore, it takes advantage of some intrinsic characteristics of public transport systems.

2. PROBLEM DEFINITION

In the most general way, the problem of interest can be defined as follows. Given: a) a set of stop locations (bi-dimensional point), b) a set of route profiles (sequence of consecutive stops), c) a set of GPS points sequences (sequence of bi-dimensional points), and d) a high resolution navigation network (bi-dimensional directed graph) find a location in the network for each stop (link), and a path in the network for each route (connected sequence of links). Figure 1 illustrates the problem by providing an example of the available input information and correct output. This problem had to be solved for the routing of the Singapore public transport bus system on a high resolution navigation network as an example of the generic issue.

2.1 Input information

The General Transit Feed Specification (GTFS) is a recent but already widely used format for specifying public transport systems. It was created by Google for feeding its geographic information applications. As of April 2011, the Singapore public transport system features 4584 bus stops, 355 bus lines where each line has several routes, e.g. different outward and return routes due to one-way streets and different coverage of serviced bus stops for weekdays and weekend. For each bus stop, GTFS provides a name and a location. A name is given for every line, and for each route it is known the line to which it belongs, the sequence of stops and a shape (sequence of GPS points) are optional. On the other hand, the high resolution navigation network is provided by NAVTEQ. It is essentially a directed graph and it is intended to represent each street in Singapore as a link. For each link origin and destination nodes, street name, number of lanes, length, flow, free speed, and capacity is provided. A bi-dimensional position is given for each node. This network has a total of 79835 links and 43118 nodes.

![Figure 1. Generating a sequence of links from a route profile and GPS points](image)

2.2 Special restrictions

There are some intrinsic characteristics of the public transport systems that should be considered as
hard restrictions to the problem. First, when a certain stop is assigned to a link in the network, this link should belong to all paths of the routes to which this stop belongs. In other words, the stop-link relationships are fixed for resolving the remaining routes; these relationships must be taken as further input, and if it is needed to change one stop for a particular route, all other routes including this stop must be resolved again. Hence, the order in which the routes are resolved is important.

Another aspect is that in Singapore’s case, although many lines are serviced in two directions with most bus stops having a corresponding stop in the opposite direction (stop located in the other side of the street), these properties can not be used because links defined by each opposite route are different (directed graph), locations of opposite stops are not necessarily perpendicularly opposite to a certain link and finally opposite direction routes not always use exactly the same streets.

However, some routes of the same line have an inclusion relationship: In peak hours, parts of bus routes facing high demand are supplemented with additional bus services running on only partial routes to meet the demand. In these cases if a full route is resolved the solution(s) of its partial route(s) is(are) included in its solution.

3. SOLUTION APPROACH

Analyzing the quality of the given GPS points in terms of the resolution (temporal resolution unknown, with a distance between consecutives points of about 65 meters) shows that it is not possible to apply advanced methods for automatic map-matching, which need usually at least 10 GPS points for each link (average length of the links is 91 meters). Furthermore, the fact that not all the routes have GPS points inhibits using a full automatic solution (38 bus routes do not have GPS points).

Hence, the strategy for resolving each route consists of a semi-automatic procedure. Figure 2 illustrates the process. First, a simple map-matching algorithm is applied if the route is not subject to the inclusion relationship described above. In these cases, only a part of a previous solution is needed to obtain a first solution. Then, an automatic verification described below is performed. If the verification ends with a positive outcome, one can decide to finish the route and save the solution or to continue editing. Otherwise, it is mandatory to make changes. If one decides, or is forced, to modify the solution there are two ways: changing parameters and running the automatic algorithm again, or editing the solution interactively with a graphical editing tool. In both cases the automatic verification must be executed again. If previously saved stop-link relationships are modified, prior routing solutions which contain one of the involved stops are erased.

As more solutions are obtained, it is becoming increasingly easier and faster to solve further routes, as in any learning process. This happens for two reasons, first for the inclusion relationships that omit the algorithm, and second because the increasing number of fixed stop-link relationships relaxes the algorithm, as explained in the next section.

![Figure 2. Semi-automatic process for one route](image-url)
4. AUTOMATIC MAP-MATCHING ALGORITHM

The objective of this algorithm is to generate a solution (path or sequence of connected links of the network, and a set of stop-link relationships) for one route, knowing its profile, a sequence of GPS points, and a set of stop-link relationships. The algorithm is designed to deal with: a) low resolution of GPS points, b) sporadic low network spatial resolution, c) long distances between two stops in the case of express routes and d) The fact that not always the nearest link to a stop point is the correct one.

The route map-matching process is illustrated in Figure 3. Except for the first stop, the algorithm solves for each stop in the route profile a portion of the link sequence (from the previous to the current one) and if this stop has no fixed link, a set of link candidates are polled from which one link is selected.

Definition of link candidates is performed as follows: the \( N_i \) closest links to the stop point, within a distance \( D_{max} \), define the set of candidates. Each set’s element could be subjected to more restrictions: the nearest point between the stop point and the infinite line defined by the link, must be inside its line segment, and the angle between the link direction and the nearest GPS points sequence direction must be lower than \( \alpha_{max} \).

Link’s selection is performed as follows: from the previous stop link to each defined candidate an A* search algorithm is applied for finding the shortest path. For this algorithm each link costs include the travel time and the distance to the GPS points. A product was defined due to the differing units between the two quantities:

\[
C_{link} = (L_{link}/S_{link})^{w1}(D_{GPS})^{w2}
\]

(1)

where \( L_{link} \) is its length, \( S_{link} \) is its free speed, \( D_{GPS} \) is its distance to the GPS points sequence, and \( w1 \) \( w2 \) are positive weights with a standard value of 1, but modifiable by the user according to the existence or quality of this GPS points sequence on a certain route. The definition of \( D_{GPS} \) can be modified, in the simplest approach it is the minimum distance between the link and all the GPS points (point-segment distance). Then, the path with the lowest length is part of the solution, and the correspondent candidate link is related to the stop.

If the current stop had a stop-link relationship only the shortest path to this stop defines the solution, and the process continues with the next stop in the route profile. If the first stop of the profile has no fixed link a similar algorithm between the first and the second stop is performed. The definition of candidates procedure is applied to the first and the second stops. Then, the candidates’ selection procedure consists of obtaining all the shortest path combinations between the two set of candidates, and selecting the one with the lowest length. This path defines link for both stops.

![Figure 3. Map-matching algorithm](image-url)
5. AUTOMATIC VERIFICATION

In this step, the correctness of the routing solution is automatically checked by performing the following, ordered verification: 1) is the path joined?, 2) is the path without U turns? 3) is the path without repeated links? 4) has every stop of the route a stop-link relationship? 5) is every stop related to a link inside the path? 6) is the order of the related links in the path the same than the order of the correspondent stops in the route profile? 7) is the nearest point between the stop point and the infinite line defined by the link inside its line segment in every stop-link relationship? And 8) are the first and last links of the path related to the first and last stops of the route profile?.

Verifications 2), 3) and 7) are not mandatory and can be deactivated through the user interface. The performance of those tests need user input as on the one hand covers possible errors, but on the other hand may include actual route characteristics: some bus routes indeed make U turns, some repeat exactly the same street in the same direction during their travel, and the geometric restriction presented in 7) is not always valid in big stop facilities like bus interchanges.

6. MANUAL EDITING FUNCTIONALITIES AND IMPLEMENTED SOFTWARE

The objective of the edit functions is to allow the user to modify the automatically generated routing solution. Even if the automatic algorithm generates a correct solution based on the input data, problems as recent changes in routes, differences in the release date between GPS points and network data, erroneous GPS points, or lack of network elements can cause needs for manual changes. Although one also could modify and correct the input data or the generated solution with direct data modifications, bi-dimensional visualization and keyboard-mouse user interaction are two quality attributes which help to reduce time and effort. Developed functionalities are described as follows:

(a) Visualization: A navigation network is displayed, including all relevant information for working with one single route. This includes the route’s profile, the given sequence of GPS points and its current solution (path and stop-link relationships). Selected elements are drawn in a different color. All is displayed in a bi-dimensional interactive way with refresh of the cursor location in the working coordinates, and panning, zoom and view-all options.

(b) Selection: Different options for selecting elements of the solution or elements from the network are provided. It is possible to select the nearest link (solution or network), nearest node (network) or nearest stop (solution) to a point indicated by the user. When a stop that has a stop-link relationship already, the corresponding link is selected as well. If a link of the solution path is selected and it does not have a subsequent link connected, a new one from the network is selected with one click; the selected link is the one with the most similar angle than the line defined by the end node of the initial link and a point indicated by the user.

(c) Path modification: The first link of the sequence can be added selecting any link of the network. If a link of the solution path does not have a subsequent link connected, it is possible to add one according to the selection function described in (b). If there are two subsequent links in the solution that are not connected (a gap), a subsequence that connects the mentioned links is added, using the shortest path algorithm, with the current parameters. Furthermore, selecting one link of the path, it is possible to delete it, or to delete all the links before or after it. Finally, stop-link relationships can be modified selecting both elements. If the modified relationship was fixed, the route solutions to which the selected stop belongs will be erased.

(d) Network modification: New nodes to the road network can be added. In addition, with any node selected, it is possible to add a new link selecting the end node.

These functions were implemented in a software developed from scratch in java and using the java2D library for graphics. It also reproduces the solution approach, traversing routes without a current solution, and running the map-matching algorithm and the automatic verification for each one. Figure 4 shows the user interface and a demo video can be accessed at http://www.vimeo.com/27137889.
7. RESULTS FOR THE SINGAPORE APPLICATION

As mentioned above the objective of the semi-automatic process described above was to solve the map-matching problem for the public bus system of Singapore. In a period of ten working days 1959 routes from 355 lines were solved defining 4584 stop-link relationships. Usage, analysis, details and publications of these results are summarized below.

7.1 MATSim simulation

MATSim is a multi-agent, activity-based transport simulation model, that has recently been expanded to include public transport (Rieser, 2010). The main aim to solving this problem was to feed several MATSim simulation scenarios for Singapore. One of the scenarios was defined for processing only public transport vehicles and other with private vehicle as well. A video of a simulation of bus vehicle agents in Singapore at the morning peak can be accessed at http://www.vimeo.com/25122496.

7.2 Time measurement of solving process

The total time of the work process is given by the sum of the algorithm computation and manual corrections. The time intensity of both task is dependent on the number of link candidates that is taken into account when mapping the bus stops to the road network. Figure 5 shows measurements of the process time (algorithm, manual and the sum) for different number of link candidates for three routes. Although computation times increase exponentially, the total time for the routing process is very much determined by the manual work. The optimal number of candidates depends on the typology of the route (length, network density, etc.) but the three trial measurements indicate that the optimal number of candidates is around two to four.

---

Figure 5. Processing time measurements for three example routes
7.3 Procedure evaluation

Some aspects can obviously be evaluated and improved. The most critical point is the ability of the automatic map-matching algorithm to generate better solutions without previous information. Analyzing its steps, when a new link is selected for a certain stop, only the subsequence starting from the previous to the given stop is taken into account. This is a problem if the selected link is inadequate for the next subsequence. Hence, the algorithm can be improved by adding roll back conditions. This will negatively affect the computation time, but it is expected that less manual work is needed at the same time. As it determines the time intensity of the process, this strategy should reduce the total time.

8. CONCLUSION AND OUTLOOK

The semi-automatic procedure designed for map-matching bus lines with a high resolution navigation in Singapore was successful, allowing to solve all bus routes and stops in only ten days. This is taking into account the quality of the input information offered, highlighting the low spatial and temporal resolution of the GPS points given for each route. Analysis of the process reveal that reducing the manual modification time is the best manner to improve the procedure. It can be done modifying the automatic algorithm for obtaining more accurate results for the initial routes to be solved, or in other words, for routes not affected by the learning process. One still has to consider the development time of such an improved algorithm before engaging in this task. As GTFS is becoming so popular for defining public transport systems and the code, in which this process has been implemented, is open source, it can be used for matching routes with high resolution networks of any GTFS-specified place. The code is available in the contrib project of MATSim. Actually, for generating MATSim simulation scenarios the process is now being used by research teams in London, UK and Gauteng, South Africa.

REFERENCES