# Benchmarking Trajectory Matchers with SUMO

Dr. Jakob Erdmann
Dr. Rüdiger Ebendt
Institute of Transportation Systems
German Aerospace Center
12489, Berlin
Germany
E-mail: Jakob.erdmann@dlr.de

**KEYWORDS**

Trajectory Matching, Simulation, SUMO

**ABSTRACT**

The trajectory matching problem refers to matching vehicle tracking data (trajectories) to the sequence of road segments taken by the vehicle driving through a road network. In order to test and compare algorithms which attempt to solve this problem, a ground truth is needed. We explain how the the software package SUMO can be used to create synthetic trajectories which are suitable to perform such evaluations.

## INTRODUCTION

A vehicle trajectory typically consists of a time-stamped series of GPS-Positions. Inferring the route of a vehicle (the sequence of road segments) from such a trajectory is a prerequisite for obtaining useful traffic data relevant to a road network (Brockfeld et al. 2007, Alt et al. 2003). The GPS positions are affected by errors and noise (caused by e.g. clouding or multi-path signals), known as the measurement error. For each position in a trajectory, trajectory-matching needs to decide onto which of several eligible candidate segments the position should be projected (Chawathe 2007). It also needs to handle the sampling error: depending on the signal frequency, the vehicle may pass one or more intermediate segments between every two reported positions. The lower the frequency, the more passed segments may exist between any two GPS positions, and map-matching needs to determine these segments.

To evaluate a trajectory matching algorithm, it is necessary to know the correct routes for a representative, preferably large, sample of trajectories. While it is certainly favorable to be as close to reality as possible, using real world trajectories of actual vehicles travelling in a city also has a disadvantage: due to the (typically) low frequency of the position reporting it is sometimes not possible to infer the correct underlying route with certainty, even when using human aid. A possible way of overcoming this has been outlined (Lou 2009): here, synthetic data has been used as a known ground truth. By starting with a synthetic route through the road network and generating a synthetic trajectory, the exact correspondence between trajectory and network is known. Moreover, this way the derived benchmarks are based on really large amounts of tracking data, rather than on small data sets from user-contributed GPS traces such as. OpenStreetMap tracking data (OSM 2008), or from a few, costly field campaigns.

In this work we explain how the software package SUMO can be used to obtain synthetic trajectories with many desirable properties. We also report on our findings from comparing three trajectory matching algorithms which were conducted within the EU-funded FP7 project SimpleFleet.

SUMO (short for **S**imulation **o**f **U**rban **M**obility) is an open source software package for the microscopic simulation of road traffic (Behrisch 2011). It allows the simulation of all vehicles within a large city and also supports public transport and pedestrian traffic. Since its initial release in 2001, SUMO has become a popular tool for traffic research (Krajzewiz 2013). SUMO comes with a host of applications which aid in the preparation of a simulation scenario such as *netconvert* for importing various digital networks and *duarouter* for computing shortest paths. This paper may be of general interest to SUMO users because it explains how several of these applications are used in concert.

## TRAJECTORY MATCHING ALGORITHMS

The algorithms compared in this work are listed in the following. The implementation details of these algorithms are outside the scope of this work.

1. **STM**: The Spatio-Temporal Matching algorithm is described in (Lou 2009). The open source implementation available at (TTA) was used in this work.
2. **BPSW**: The algorithm described in (Brakatsoulas, Pfoser, Salas and Wenk 2005) was not directly available. Instead, trajectories were sent to a project partner and routes (lists of segment ids) were later retrieved from a database server
3. **DLRM**: The DLR-Matching Algorithm is described in (Ebendt et al. 2010a, 2010b, 2012). An in-house implementation was used.

## NETWORK PREPARATION

All trajectory matching algorithms receive as input a trajectory and a digital road network. These road networks are represented by a graph made up of nodes and edges. We use the terms road segment, segment and edge

interchangeably. The output of a matching algorithm is a list of edge ids called route.
To obtain an accurate comparison between two algorithms, it is imperative that they both receive the same road network as input. Differing networks might change the problem (i.e. if one network has a higher density of alternative routes) and they would also make it hard to compare returned routes by counting correctly matched edges (i.e. if one network divides the network into a larger number of shorter edges).
Because all algorithms listed in the previous section use different network formats, it was necessary to find a common network definition and transform it into the respective formats. The *netconvert* application included with SUMO facilitates this work by providing a large number of input- and output-formats for network generation. Another important feature of *netconvert* is the heuristic generation of traffic light programs for all nodes which are marked as being controlled by a traffic light system. This ensures that trajectories feature realistic stopping behavior. The network conversions are described in the following.

**Conversion of BPSW network for use by DLRM and SUMO**

The network data used by the BPSW algorithm was based on OpenStreetMap data of the city of Berlin. Some transformations to turn it into a routing graph were applied to it by another research group and it was only accessible via a read-only data-base connection within the scope of this work. Thus it was necessary to obtain an equivalent network in a format compatible with the DLRM algorithm. First, the graph of nodes and edges was retrieved from the data-base and transformed into the *plain-xml* format used by *netconvert* (SUMODOC). It was found that the BPSW network data uses only straight edges and thus employs a very large number of edges to represent curved geometries. Because *netconvert* enforces a minimum length requirement of 10cm per edge, some short edges had to be removed by joining them with longer edges to meet this constraint and maintain the network topology. This affected 106 edges out of a total of 98670 edges and is deemed irrelevant to the evaluation (removed edges in the returned route were simply ignored). The resulting network data in plain-xml format was then converted into two other formats using the *netconvert* application:

- A SUMO simulation network (.net.xml format) for generating synthetic trajectories via micro-simulation
- A variant of the GDF format (ISO 14825:2011) which is used by the DLRM algorithm

Thus, both matching algorithms BPSW and DLRM as well as the SUMO simulation were able to run on equivalent networks.

**Conversion of STM network for use by DLRM and SUMO**

For the comparison of STM and DLRM, the same geographic region as in the previous section was chosen. The STM algorithm is part of a software package which contains a custom application (*OSM2Routing*) for importing OpenStreetMap data. This generates network data using an alternative XML format which is more suitable for routing. Unfortunately, this custom application was written when OSM-data contained integer IDs in the 32 bit range. Current OSM-data uses 64 bit IDs which necessitated a remapping of all IDs into the 32 bit range to allow usage of the application. The output data of *OSM2Routing* could be transformed into the plain-xml format used by *netconvert* with little effort. As in the previous section, the resulting network data in plain-xml format was then converted into the two other formats used by the SUMO simulation and the DLRM algorithm. Thus, both matching algorithms STM and DLRM as well as the SUMO simulation were able to run on equivalent networks.

**SYNTHETIC ROUTES**

Generating synthetic routes was performed in two steps. First, demand relations consisting of a start edge and a destination edge were generated using the SUMO tool *randomTrips* for the networks described in the previous sections. A total number of 10000 such trips were generated with a minimum beeline distance of 1km. In a second step, these trips were transformed into routes with a fastest-path routing algorithm (Dijkstra 1959). Two different sets of routings were performed using the *duarouter* application included with SUMO:

- Empty network: for each trip, the fastest route through the network was selected according to the speed limits found in the network data. This is a plausible scenario for night-time driving where vehicles are barely obstructed by other vehicles.
- Congested network: for each edge in the network, a reduced speed was computed by multiplying the speed limit with a random factor *r*. The factor *r* was sampled by computing $1 - |N(0,0.5)|$ and truncating to $[0.2, 1]$. This was meant to represent obstructions in a congested network. According to these speed reductions, alternative simulation networks to those describe in section 1.4 were generated where the speed limit of each edge was reduced. This was done by altering the generated plain-xml input files. The fastest route in these networks was then used for each vehicle.

**TRAJECTORY GENERATION**

The routes generated in the previous step were loaded along with the generated sumo.net.xml into the SUMO simulation. The following simulations were performed:

1) BPSW-derived empty network with fastest routes
2) BPSW-derived congested network with fastest routes based on the modified speeds
3) STM-derived empty network with fastest routes
4) STM-derived congested network with fastest routes based on the modified speeds

Simulations 1) and 3) correspond to situations in which the drivers are aware of the network being empty whereas simulations 2) and 4) correspond to drivers knowing about the congestions and selecting their routes accordingly. In these cases, the speed reductions in the network corresponded exactly to those used during routing. An alternative combination might also be relevant but was not pursued in this work: Routes computed for the empty network might be used in a simulation with reduced edge speeds corresponding to drivers with imperfect information.

Each of the simulations was run with the option *–fcd-output* which causes vehicle trajectories with perfect spatial accuracy and a temporal resolution of 1 second to be written. Additionally, the option *–fcd-output.geo* was used to receive output in geodetic instead of Cartesian coordinates. The following transformations where then applied to the simulation outputs to create input files for the matching algorithms:

- Sampling of data points with a fixed frequency of 30 seconds
- Spatial distortion of the GPS-positions by a 2-dimensional normal distribution with standard deviation of 5 meters.
- Transformation into a column oriented format called gpsdat for BPSW and DLRM
- Transformation into an xml format called GPX for STM

Modelling the horizontal error distribution of the GPS positions, i.e. the distribution in the x-y plane, by a 2-dimensional normal distribution has been widely adopted in the related research field (Lou 2009, Pfoser 1999). All of the above transformations are supported by the SUMO tool *traceExporter*. Further supported output formats can be found in (SUMODOC2). Note that the generated trajectories are influenced by the traffic lights in the simulation network which serves to increase their realism.

## EVALUATION RESULTS

Due to the difficulties involved in obtaining equivalent simulation networks for use with the matching algorithms, direct comparisons were only performed between the pairs of (STM, DLRM) and (BPSW, DLRM). This decision was influenced by the existing ability of generating networks for use with DLRM. Nevertheless, the obtained ranking should be sufficient to draw conclusions regarding the relative strengths of STM and BPSW. It must be stressed that algorithms working on individual graph edges cannot be directly compared if the graphs differ even if the different graphs are based on the same geographic region. The trajectories from each of the four simulations described in the previous section were used as input for two different matching algorithms giving a total of 8 result sets. In the following we will first discuss the used evaluation metrics and then present the results.

### Evaluation Metrics

To evaluate the quality of a trajectory matching result, the known route (as an ordered list of edge ids) is compared against the map-matched route (again, an ordered list of edge ids). As an example, consider the original route [A,B,C,D,E,F] and the map-matched route [A,B,E,D,G], then the comparison accounts for the following deviations:

- The map-matched route does not contain all edges (C and F)
- The map-matched route contains the original edges, but in the wrong order (E, D)
- The map-matched route contains edges which do not occur in the original route (G)

In the following, we distinguish between target value (T), and the actual value (A). The target value equals the number of edges in the original route. The actual value equals the difference between the number of edges in the map-matched route which also occur in the original route, and the number of those edges in the map-matched route which do not occur in the original route (that is, “wrong” edges in the result act as an arithmetical penalty). Notice that, by construction, duplicate edges do not occur in an original route. They may however occur in a map-matched result route (and, in this case, the existence of such edges means that the penalty value is increased accordingly).
Then, a first quality metric Q for a particular route is given by

$$Q = A/T \cdot 100\%$$

This metric is called “accuracy by number”. Three more variants of this metric have been calculated:

- Accuracy by length: T is replaced by the sum of lengths of edges in the original route, whereas A is replaced by an analogous difference of sums of edge lengths instead of a difference of edge numbers. This can be thought of applying the lengths of the edges as their weight in a weighted sum.
- Accuracy by number, ordered: as before with accuracy by number, but now edges in the map-matched route add to an increase of A only when they occur in the same order as in the original route.
- Accuracy by length, ordered: when building the sums of edge lengths for the calculation of A, only edges are considered which occur in the same order in the original route.

For all variants, cumulated or aggregated metrics can be defined for the whole set of considered routes. For this purpose, cumulative values

$$T = \sum T_i \, , \; A = \sum A_i$$

are calculated, summing up all target values $T_i$ and all actual values $A_i$ for all routes *i*. Then, the cumulated quality Q is calculated as before.
A disadvantage of this metric is that the dependency between $T_i$ and $A_i$ as the two key figures describing the matching quality for one particular route is lost completely.
Alternatively, the matching accuracies for the individual routes can be aggregated, that is, mean or median values can be calculated. That way, the aforementioned dependencies are not destroyed. The median value has a certain advantage over the mean value since the median value tends to be more robust with respect to outliers.
For these reasons, we prefer giving the median values of the observed individual matching accuracies. Nonetheless, for completeness sake, also the cumulative values have been calculated, and are presented in the following.

### BPSW versus DLRM

Figure 1 shows the distribution of Q values for individual trajectories. Table 1gives a comprehensive list of performance metrics for simulations 1) and 2) for both algorithms. It can be seen that DLRM performs better than BPSW by a margin of 20-28% depending on empty or congested conditions. It can also be seen that the difference between the algorithms is hardly affected by the different quality metrics.

The low quality of the BPSW results for some trajectories could in part be due to implementation errors rather then errors in the algorithm itself. Even though an analysis of the implementation was outside the scope of this work, a brief inspection revealed gaps in the matched routes which had unique paths between the correctly matched edges and thus could have been avoided trivially.
In the congested network BPSW appears to perform better than in the empty network at least when coutning the number of matched edges. This performance gain vanishes when using the metric *by length*. No explanation for this effect can be given at this time..

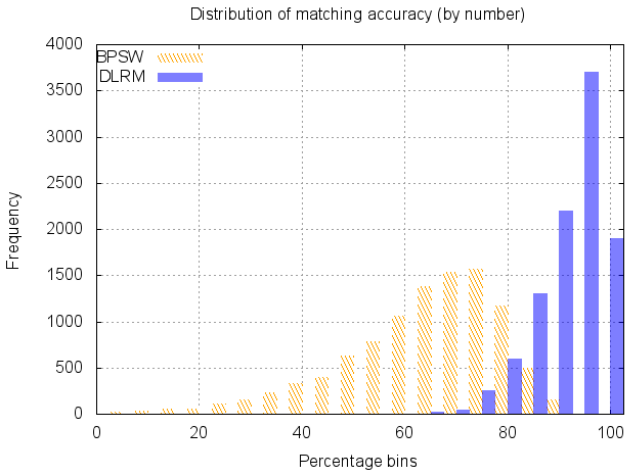


Figure 1 Histogram for the metric **accuracy by number** for algorithms STM (orange) and DLR (blue) for the trajectories from simulation 2 (empty)

Table 1 Performance metrics for comparing BPSW and DLRM. Median values (med) and cumulative values (cum)

| | Q by number | | Q by number (ordered) | | Q by length | | Q by length (ordered) | |
|---|---|---|---|---|---|---|---|---|
| | med. | cum. | med. | cum. | med. | cum. | med | cum. |
| BPSW empty | 66.9 | 63.6 | 66.7 | 63.3 | 67.9 | 65.4 | 66.7 | 65.2 |
| DLRM empty | 93.4 | 91.4 | 93.4 | 91.8 | 94.5 | 91.4 | 94.5 | 91.8 |
| BPSW cong. | 71.8 | 70.4 | 71.8 | 70.3 | 68.8 | 66.4 | 68.7 | 66.3 |
| DLRM cong. | 92.0 | 89.9 | 92.0 | 89.9 | 93.2 | 90.4 | 93.2 | 90.4 |

### STM versus DLRM

Figure 2 shows the distribution of Q values for individual trajectories. Table 2 gives a comprehensive list of performance metrics for simulations 3) and 4) for both algorithms. It can be seen that both algorithms achieve a high level of accuracy with STM performing better than DLRM by 3-10% depending on empty or congested conditions.

[Geben Sie Text ein]

Again, it can be seen that the difference between the algorithms is hardly affected by the different quality metrics.

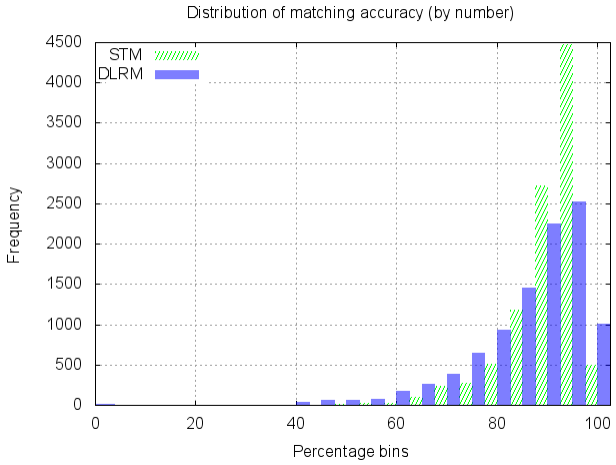


Figure 2 Histogram for the metric **accuracy by number** for algorithms STM (green) and DLR (blue) for the trajectories from simulation 4 (empty)

Table 2 Performance metrics for comparing STM and DLRM. Median values (med) and cumulative values (cum)

| | Q by number | | Q by number (ordered) | | Q by length | | Q by length (ordered) | |
|---|---|---|---|---|---|---|---|---|
| | med. | cum. | med. | cum. | med. | cum. | med | cum. |
| STM empty | 92.3 | 91.7 | 92.3 | 91.7 | 92.6 | 91.6 | 92.3 | 91.6 |
| DLRM empty | 89.4 | 86.6 | 89.4 | 86.5 | 92.0 | 87.8 | 92.0 | 87.7 |
| STM cong. | 93.6 | 93.3 | 93.6 | 93.3 | 94.3 | 93.7 | 93.6 | 93.7 |
| DLRM cong. | 86.8 | 82.2 | 86.8 | 82.1 | 89.5 | 83.7 | 89.5 | 83.6 |

## CONCLUSION

It was shown how the various applications in the SUMO suite can be used in concert to create synthetic trajectories for the task of benchmarking trajectory matching algorithms. Three matching algorithms were compared and a relative ranking was obtained (from best to worst: STM, DLRM, BPSW). We note, that the large number of idiosyncratic network formats currently in existence and the fact that each of the trajectory matching algorithms requires a different network format complicates their comparison. While support for the OpenStreetMap format appears to be growing, it does require some transformations to turn it into a routing-graph and thus does not stop the proliferation of new formats.

## ACKNOWLEDGEMENT

This work was partially supported by the European Commission through its Seventh Framework Programme

during the project ``SimpleFleet" (http://www.simplefleet.eu, grant agreement No. FP7-ICT-2011-SME-DCL-296423).

## REFERENCES

Alt, H.; Efrat, A.; Rote, G.; Wenk, C. 2003: Matching planar maps. J. of Algorithms, 49:262–283.

Behrisch, Michael; Bieker, Laura; Erdmann, Jakob; Krajzewicz, Daniel, 2011, "SUMO - Simulation of Urban MObility: An Overview". In: SIMUL 2011, The Third International Conference on Advances in System Simulation, pp. 63-68, 2011.

Brakatsoulas, S; Pfoser, D.; Salas, R.; Wenk, C., , 2005: On Map-Matching Vehicle Tracking Data. In VLDB, pp. 853-864.

Brockfeld, Elmar; Wagner, Peter 2007; Passfeld, Bert: Validating travel times calculated on the basis of taxi floating car data with test drives. In: ITS World Congress, Beijing.

Chawathe, S.S. 2007, Segment-based map matching. In: Proceedings of the IEEE Intelligent Vehicles Symposium, Istanbul, pp. 13–15.

Dijkstra, E.W. 1959, A note on two problems in connexion with graphs. Numerische Mathematik 1, , pp. 269–271.

Ebendt, Rüdiger; Sohr, Alexander; Touko Tcheumadjeu, Louis Calvin; Wagner, Peter, 2010: Utilizing historical and current travel times based on floating car data for management of an express truck fleet. In: 5th International Scientific Conference: Theoretical and Practical Issues in Transport, Pardubice, ISBN 978 80 7395 245 7.

Ebendt, Rüdiger; Wagner, Peter 2010: An Integrative Approach to Light- and Heavy-Weighted Route Planning Problems. In: 5th IMA Conference on Mathematics in Transport, London.

Ebendt, Rüdiger; Sohr, Alexander; Touko Tcheumadjeu, Louis Calvin; Wagner, Peter 2012: Dynamische Neuplanung der Touren von Express Trucks unter Einbeziehung einer FCD-basierten Verkehrslage. Multikonferenz der Wirtschaftsinformatik (MKWI 2012), Braunschweig, ISBN 978 3 942183 63 5, 2012.

Krajzewicz, Daniel (2013) Summary on Publications citing SUMO 2002-2012. In: 1st SUMO User Conference - SUMO 2013, 21, pp. 11-24. DLR. SUMO2013 ISSN 1866-721X

Lou, Y.; Zhang, C.; Zheng, Y.; Xie, X.; Wang, W.; Huang, Y.,2009, Map-Matching for Low-Sampling-Rate GPS Trajectories. In ACM GIS, pp. 352-361.

Pfoser, D.; Jensen, C.S, 1999: Capturing the Uncertainty of Moving-Object Representations. Proceedings of the 6th International Symposium on Advances in Spatial Databases, Springer-Verlag London, UK, pp. 111-132.

Mordechai M. Haklay, Patrick Weber, 2008, OpenStreetMap: User-Generated Street Maps IEEE Pervasive Computing, Vol. 7, No. 4. (2008), pp. 12-18, doi:10.1109/mprv.2008.80

## WEB REFERENCES

SUMO sumo-sim.org

SUMODOC sumo-sim.org/userdoc/Networks/Building_Networks_from_own_XML-descriptions.html

SUMODOC2 http://sumo-sim.org/userdoc/Tools/TraceExporter.html

OSM www.openstreetmap.org

TTA code.google.com/p/traveltimeanalysis