ETH zürich

Implementation and Evaluation of Trajectory Similarity Measures

Sven Ruf
Bachelor Thesis, FS 2020
Supervisors: Jannik Hamper, Dominik Bucher, Prof. Dr. Martin Raubal

IKG
Institut für Kartografie und Geoinformation

geoinformation engineering
ikg eth zurich

1 Introduction

Trajectory similarity measures are implemented and evaluated within the context of the mobility and movement processing framework ***trackintel***. This python framework is developed by the MIE-lab at ETH Zurich. It offers workflows to preprocess and analyze movement data. A module for similarity measures is added to the framework.

2 *trackintel* Architecture

The existing methods of ***trackintel*** offer preprocessing methods to extract triplegs, staypoints, etc. according to the data model (figure 1). The new similarity module offers the following methods:

1. Dynamic Time Warping (DTW)
2. Edit Distance on Real Sequences (EDR)
3. Start End Distance
4. Similarity Matrix Calculation

Methods 1 and 2 are existing similarity measures, that are adapted to work within ***trackintel***. Method 3 is an own approach best suited for individual transport data. Method 4 offers the user the possibility to calculate the trajectory distances or similarities (inverted trajectory distances) for a whole data set.

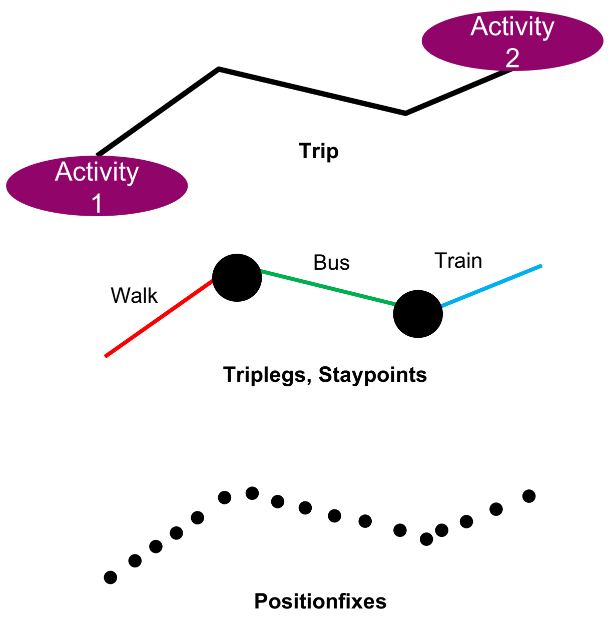

Fig. 1 Data Model of ***trackintel***

3 Start End Distance

The similarity measure Start End Distance, that is introduced, is created to cover situations where similarity is not dependent of the route between origin and destination. An applied example is ridesharing, where two people can share a ride when going from the same start to the same end point, no matter which road they would take in their own car. Such a situation is showed in figure 2.

The Start End Distance is calculated as a weighted sum of the distances and time differences at start and end of a trajectory to the nearest point of another trajectory. The user can define a weighting function to set the influence that the distances and time differences at the start and the end have.

The algorithm compares one trajectory (T_A) with all others (figure 3) and calculates the according Start End Distance. To determine all possible candidates the user defines a time and a distance threshold. The trajectories are filtered using a buffer around the start and end of T_A. If also the time differences are below the time threshold, the trajectory is a candidate and the Start End Distance is calculated. In figure 3 this would only be T_1.

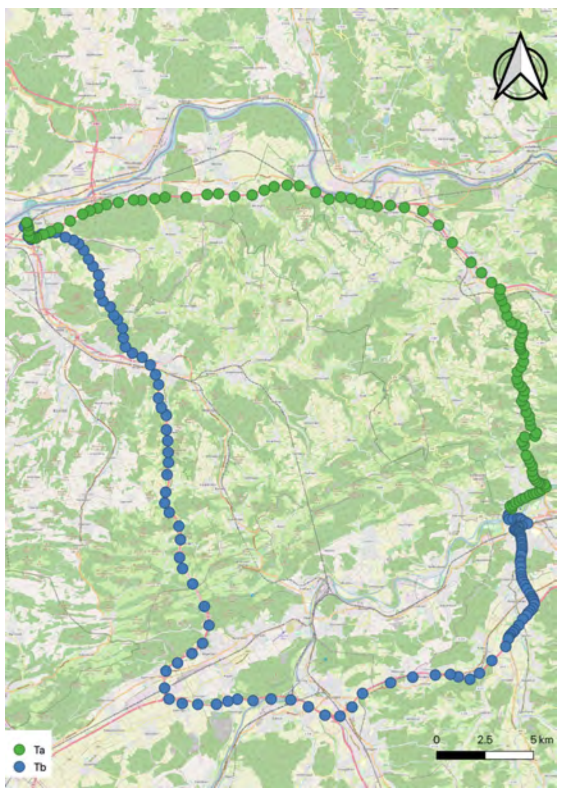

Fig. 2. Two Trajectories with Same Origin and Destination

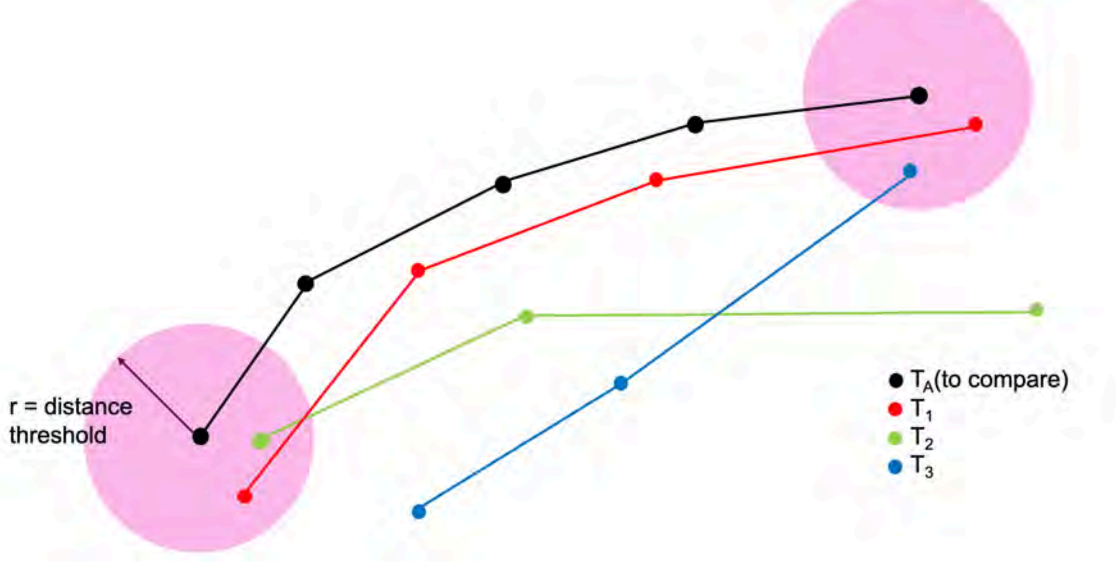

Fig. 3. Filter of Trajectories with Distance Buffer

4 Conclusion

The existing similarity measures and the new approach have been integrated in the ***trackintel*** framework. Various test cases and data sets have been processed. The DTW distance is directly affected by outliers and by their distance to the rest of a trajectory. EDR performs better with ouliers in the data, because the distance of trajectory points is only mapped indirectly by comparing them with a threshold. The new approach Start End Distance performs good in the cases it is designed for. But the definition of similarity is tailored for the specific use case of ridesharing and can therefore not be adapted to more general situations easily. Regarding the framework, further functionality e.g., smoothing algorithms can be integrated in the future, to increase the part of a data processing workflow, where ***trackintel*** can be used.