Using V2X-Information for Trajectory Prediction at Urban Intersections

Michael Klöppel-Gersdorf[a] and Thomas Otto
Fraunhofer IVI, Fraunhofer Institute for Transportation and Infrastructure Systems, Dresden, Germany

Keywords: Vehicle-to-Everything Communication, ITS-G5, Automated Driving, Intersections, Trajectory Prediction.

Abstract: Crossing an urban intersection is one of the major challenges in automated/autonomous driving. This is due to a manifold of possible interactions with other traffic participants. In this paper, we propose a trajectory prediction service based on historical V2X-information gathered from Cooperative Awareness Messages (CAMs)/Basic Safety Messages (BSMs). The service allows connected vehicles to more easily navigate the intersection by identifying possibly critical encounters, especially with traffic participants which are not covered by the vehicle's sensors. In comparison with approaches relying on video or other sensor data sources, this has the advantage that Road-Side Units (RSUs), which are used for Vehicle-to-Everything (V2X) communication, are more and more available at public intersections, e.g., due to equipment rollouts all over Europe in projects like C-Roads. The prediction service introduced in this paper is a first step in ongoing research and will act as a baseline for further projects, where additional sensors and also more involved prediction algorithms, e.g., based on neural networks, will be considered.

1 INTRODUCTION

Urban intersections are a hot spot for traffic accidents. In Germany, in 2021, more than 20% of accidents in urban area occurred at intersections (Statistisches Bundesamt (Destatis), 2022). This clearly implies that navigating an urban intersection is a non-trivial task for human drivers and even more so for automated/autonomous vehicles, compare for example (Cosgun et al., 2017; Hubmann et al., 2018). One major problem is the uncertainty about the other drivers' behavior. In (Hubmann et al., 2018), four key sources of uncertain future behavior are defined: unknown intention, longitudinal uncertainty, probabilistic interaction and sensor uncertainty.

In this work, we want to tackle the problem of determining the drivers' intention based on historical data obtained at an intersection. Here, the data is obtained via V2X communication, i.e., based on either CAM (ETSI EN 302 637-2 V1.4.1 (2019-04), 2019) or BSM. The drivers' intention is then determined by calculating a most likely trajectory given the past and current position of the vehicles under examination. The work is done in two parts, first clustering the historical data mainly following the approach in (Banerjee et al., 2020) and then predicting trajectories, mainly following (Wu et al., 2021). The main contributions of this paper are:

- presenting a real-time capable prediction service for connected vehicles based on V2X communication,
- using specialized distance measures between (partial) trajectories to speed up computation,
- and returning several reasonable trajectories if likely.

The remainder of the paper is organized as follows. The methodology will be introduced in Section 2, whereas results from a real world intersection will be presented in Section 3. The paper is concluded in the last section.

2 METHODOLOGY

In this section, the methodology used for clustering and prediction is described. As the discussion is mainly centered about the notion of (partial) trajectories, we start with introducing the notion of a trajectory. In general, a trajectory is considered to be a sequence of two dimensional coordinates at certain time steps, i.e,

$$T_{t_0}^{t_n} = (\vec{x}(t_0), \ldots, \vec{x}(t_n)), t_0 < t_1 < \ldots < t_n. \quad (1)$$

As we are only interested in the drivers intention, not in the dynamical behavior, the timing information is

[a] https://orcid.org/0000-0001-9382-3062

Klöppel-Gersdorf, M. and Otto, T.
Using V2X-Information for Trajectory Prediction at Urban Intersections.
DOI: 10.5220/0012735600003702

In *Proceedings of the 10th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2024)*, pages 473-479
ISBN: 978-989-758-703-0; ISSN: 2184-495X

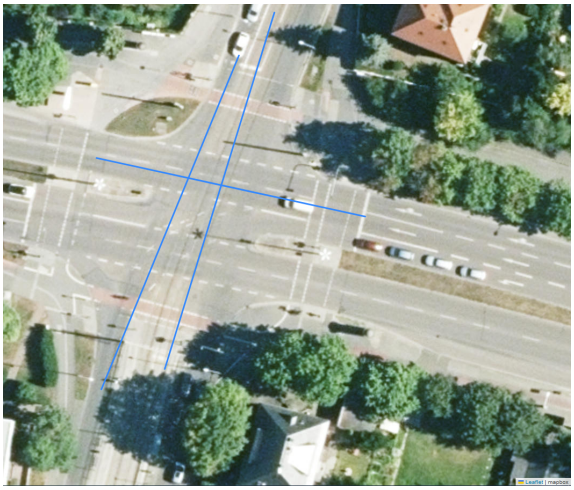


Figure 1: Overview over our reference intersection, the Fraunhofer Smart Intersection (Klöppel-Gersdorf et al., 2021), located in the city of Dresden, Germany. Shown are the principal movement vectors in approx. east-west and north-south directions. Due to the layout of the intersection, two principal vectors are required in north-south direction to successfully cluster the trajectories.

dropped. For the remainder of the paper, a trajectory will, therefore, be considered to be a sequence of two-dimensional coordinates, i.e,

$$T = (\vec{x}_0, \ldots, \vec{x}_n). \tag{2}$$

2.1 Data Acquisition

V2X information in form of the CAM is used to derive trajectories and we rely on information transmitted by commercial vehicles. The CAMs are received by a RSU installed at the Fraunhofer Smart Intersection, located in Dresden (Klöppel-Gersdorf et al., 2021). At the time of writing, about 0.5% of all vehicles at the reference intersection use V2X technology based on ETSI ITS-G5. While the CAM consists of a multitude of information (e.g. dynamic information regarding velocity and acceleration, but also the status of turn indicators), only the current position in WGS84 coordinates and the station id is used in this work. Besides the position, also bounds on the position accuracy are transmitted. The connected vehicles on the road today typically report a position error of approx. 3 m.

One specialty of the station id is that it changes every 60 seconds for privacy reasons. While it would be possible to stitch together trajectories after a change of id, at least at the low penetration rates present today, the standard discourages from doing so. Therefore, trajectories are discarded if a change of id occurs within the area of the intersection.

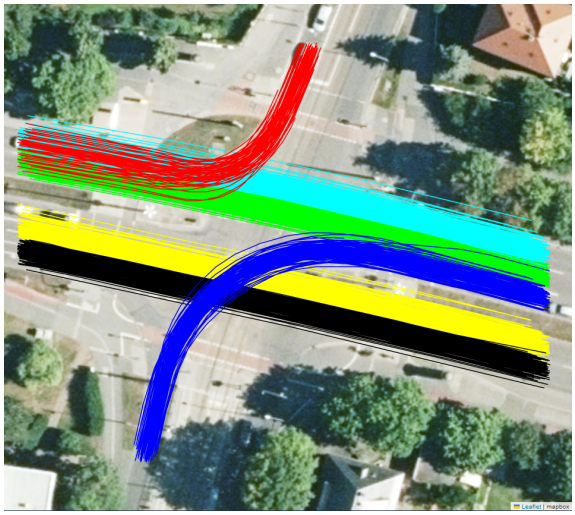


Figure 2: Part of the historical trajectories after the second step of the hierarchical clustering scheme. Trajectory clusters are now lane specific.

The trajectories used for clustering were collected in July and August 2023. The received trajectories were preprocessed, i.e., only the parts of the trajectories inside a rectangular area comprising the intersection were kept and partial trajectories (e.g., due to a change in id or repeated use of the same id) were removed. The first preprocessing step removed nearly half of the received trajectories, i.e., from 3313 initially received trajectories only 1608 vehicles actually crossed the reference intersection. Removing partial trajectories further reduced the trajectory count to 1535 trajectories, which were used for the clustering step.

As a last step, coordinates were transformed from WGS84 to an euclidic tangent plane (orthographic projection) centered at the mid of the intersection.

2.2 Clustering

For clustering the historical trajectories, a multi step hierarchical clustering scheme, similar to (Banerjee et al., 2020), is used. The first step of this scheme consists of comparing the net movement vector of the trajectories, defined by

$$\vec{m}(T) = \vec{x}_n - \vec{x}_0 \tag{3}$$

to predefined principal vectors $p_0, \ldots, p_k$, describing the straight motion over the intersection. More clearly, the cosine given by

$$\cos(\angle(\vec{m}(T), \vec{p}_i)) = \frac{\langle \vec{m}(T), \vec{p}_i \rangle}{\|\vec{m}(T)\| \|\vec{p}_i\|}, i = 0, \ldots, k, \tag{4}$$

where $\langle \cdot, \cdot \rangle$ denotes the scalar product, is used to separate straight movements from turns and also to derive

Figure 3: Historical trajectories after first step of the hierarchical clustering scheme. Straight movements are separated, but multi-lane traffic is still clustered into one class. For turn movements, two relations typically clustered together. This is due to the rather simple algorithm relying on cosine.

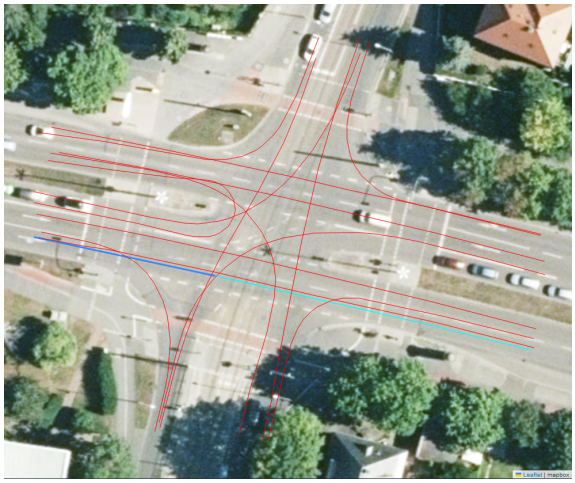


Figure 4: Reference trajectories derived for all driving relations in red. Also shown is one of the evaluation trajectories. Dark blue shows the part of the trajectory used for prediction, whereas the remainder of the trajectory is shown in cyan. This part was only used for the computation of Average Displacement Error (ADE) and Final Displacement Error (FDE).

the general direction of motion. Due to the more complicated layout of our reference intersection (tram station between the lanes on the southern part of the intersection), actually two direction vectors are required to describe the north-south direction. The three principal vectors of the reference intersection are shown in Fig. 1, whereas the result of the first clustering step is shown in Fig. 3.

In a next step, the turning movements and the streams along the two-lane streets (east-west and west-east direction) need to be separated. In (Banerjee et al., 2020) the position of the stop bar is used to differentiate the different turning movements (e.g., the green trajectories in Fig. 3). Here, a slightly different approach based on a distance measure is proposed. Let the length of a trajectory T be defined as

$$length(T) = \sum_{i=1}^{n} \|\vec{x}_i - \vec{x}_{i-1}\|. \tag{5}$$

Then, similar to (Wu et al., 2021), the distance between two trajectories is defined as

$$dist(T_1, T_2) = \frac{area(T_1, T_2)}{length(T_1) + length(T_2)}, \tag{6}$$

where the $area(\cdot,\cdot)$ is defined by the area of the polygon generated by the two trajectories, i.e., if $T_1 = \left(\vec{x}^1_0, \ldots, \vec{x}^1_{n_1}\right)$ and $T_2 = \left(\vec{x}^2_0, \ldots, \vec{x}^2_{n_2}\right)$, the area of the polygon $\left(\vec{x}^1_0, \ldots, \vec{x}^1_{n_1}, \vec{x}^2_{n_2}, \ldots, \vec{x}^2_0\right)$. Note that the coordinates of the second trajectory are reversed here. This area can be computed by geometric packages like Shapely in Python. This approach does not need Dynamic Time Warping (Kruskal and Liberman, 1983) or the variant FastDTW (Salvador and Chan, 2007) used by the authors of (Banerjee et al., 2020; Wu et al., 2021). This reduces the computation time of the algorithm significantly, because even if FastDTW is linear in time and space, it still requires a rather long computation time.

Separating the turn movements consists of a choosing a random reference trajectory inside the cluster, e.g., the first one T_1. For all trajectories in the cluster $T_1, \ldots, T_l$ the distance to the reference trajectory is computed. Trajectories following a similar movement will show only minimal distance in comparison to movements on the other side of the intersection. Both clusters can then easily be separated using a simple threshold for distance. This computation is linear in the number of elements in the cluster.

Separating the two-lane traffic uses a similar procedure. Again, starting with a reference trajectory T_1, the distances to all other trajectories in this cluster are computed. A trajectory T^{b_1} is computed as

$$T^{b_1} = \arg \max_{T=T_2,\ldots,T_l} dist(T_1, T). \tag{7}$$

A second trajectory T^{b_2} is computed as

$$T^{b_2} = \arg \max_{T=T_1,\ldots,T_l} dist(T^{b_1}, T). \tag{8}$$

Figuratively speaking, T^{b_1} and T^{b_2} are the two outermost trajectories of the cluster. As a last step, the distance of all trajectories to T^{b_1} and T^{b_2} are calculated. Finally, the trajectories are clustered depending

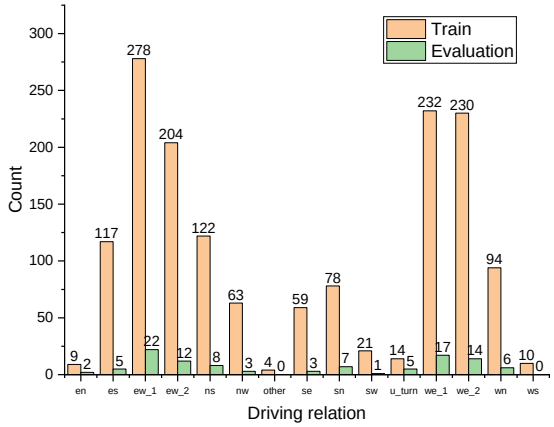


Figure 5: Distribution of the single driving movements for training and evaluation data set. A χ^2 homogeneity test revealed significant differences between the two datasets regarding the distribution of the single driving maneuvers.

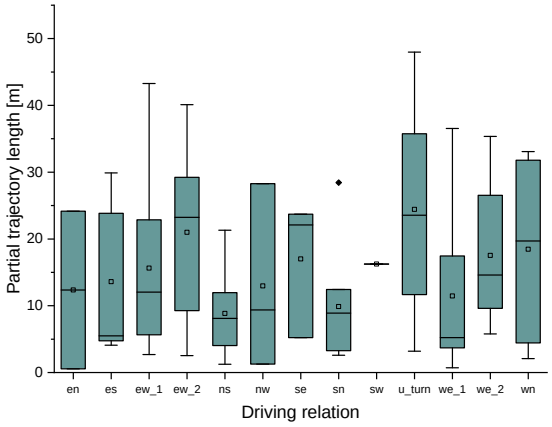


Figure 6: Box plot showing the distribution of the length of the partial trajectories per driving relation. These trajectories have a length of up to 50 m.

on whether they are closer to T^{b_1} or T^{b_2}. This process is also linear in the number of elements in the cluster, even though it uses four passes over all elements.

A part of the derived clusters are shown in Fig. 2. Please note that no spectral clustering is carried out, i.e., there is only one cluster for all movements, e.g., in west-north direction. While this limits the ability of the presented approach to detect outliers, it also reduces computation time immensely, since the computational demands of spectral clustering increase quadratically with the number of elements in the cluster.

The last step of the clustering algorithm consists of determining reference trajectories for all clusters. Like (Banerjee et al., 2020), a representative trajectory for each cluster is computed as the central element. The approach used is the same as in separating the straight movements, i.e., beginning with a random element of the cluster one outer trajectory is calculated. In a second step, the distance of all trajectories of the cluster to this outer trajectory a computed. The representative trajectory is chosen as the trajectory with the median distance. The representative trajectories for all movement relations are shown in Fig. 4.

2.3 Prediction

For predicting the future movement of a given partial trajectory (see Fig. 4 for an example), we loosely follow (Wu et al., 2021) in that we use the representative trajectories, derived above, to first calculate possible movement candidates. Instead of using a neural network, as done in the previous mentioned work, we decided to compare the partial trajectories directly to the historical data. While it may be intuitive to use the previously defined $dist(\cdot,\cdot)$ to derive the candidates, this actually proved not successful in practice. One reason for this is that some of the polygons constructed in distance calculation are self-intersecting, leading to problems of computing the area (Shapely does not guarantee correct area computation for self-intersecting polygons). Therefore, a new distance measure based on ADE and FDE was used. Here,

$$ADE(T_1,T_2) = n^{-1}\sum_{i=0}^{n} algebraicdist(\vec{x}_i^1,T_2), \quad (9)$$

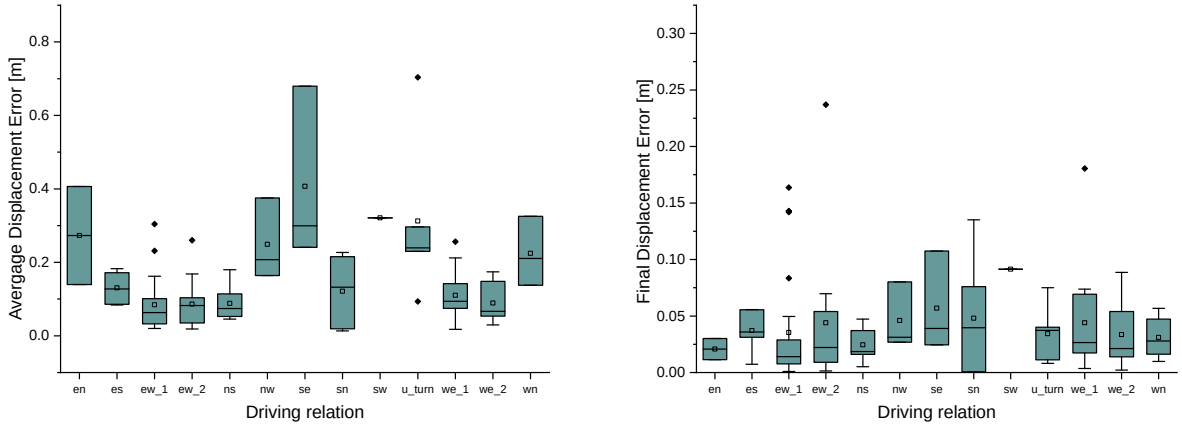
$$FDE(T_1,T_2) = algebraicdist(\vec{x}_n^1,T_2), \quad (10)$$

where $algebraicdist(\cdot,\cdot))$ is the algebraic distance of the point in the first argument to the line string in the second argument. The final distance measurement is computed as

$$weighteddist(T_1,T_2) = \alpha ADE(T_1,T_2) + (1-\alpha)FDE(T_1,T_2), \quad (11)$$

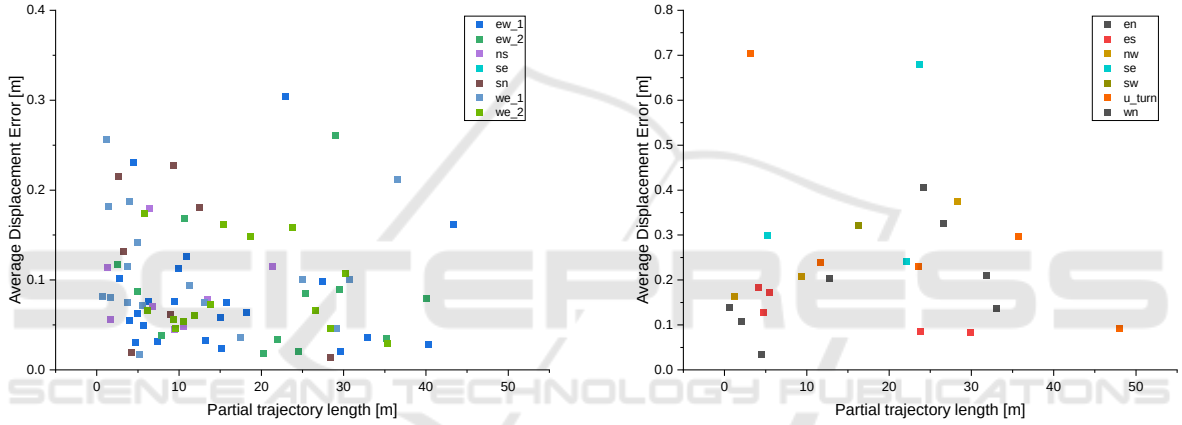
for $0 \leq \alpha \leq 1$. The reason for giving extra weight to the last point of the partial trajectory are trajectories as shown in Fig. 4. While this is most certainly a straight movement, it is still rather close to the representative trajectory of the right turn. Giving additional weight to the last coordinate increases this distance. Candidate movement relations are derived by computing $weighteddist(\cdot,\cdot)$ for the partial trajectory and all representative trajectories. Candidates are selected, if the distance is not larger than a given threshold.

In a second step, the partial trajectory will be compared against all trajectories in the candidate clusters, again using *weighteddist*. As prediction, the historical trajectory with overall the least distance is returned. The information about the possible candidate clusters is also available.



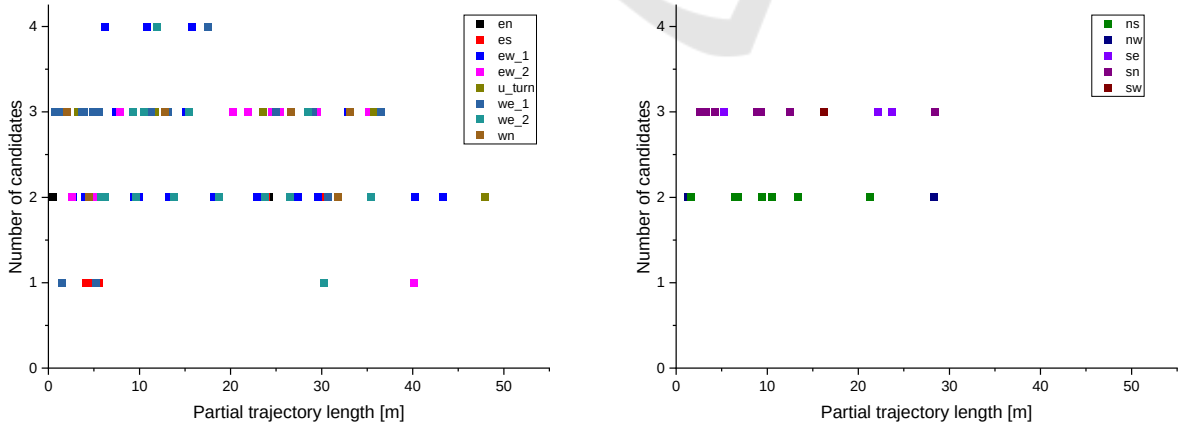
(a) (b)

Figure 7: Average Displacement Error (ADE) (a) and Final Displacement Error (FDE) (b) per driving relation.



(a) (b)

Figure 8: Length of the partial trajectory vs Average Displacement Error (ADE), for straight movements (a) and turns (b).



(a) (b)

Figure 9: Length of the partial trajectory vs. the number of potential representative trajectories, for vehicles entering the intersection from east or west (a) and north or south (b). These figures also show that we were able to generate candidate trajectories for every trajectory in the evaluation set.

2.4 Implementation

The RSU saves the position and station id of the received CAM in a comma-separated text file. These log files are gathered periodically from the RSU. Analysis is then performed with Python, mainly using GeoPandas[1] and Shapely[2]. The orthographic projection is handled by PyProj[3]. Visualizations are generated using Folium[4]. Clustering the 1535 historical trajectories takes about four seconds, predicting the 105 partial trajectories takes about ten seconds, or about 0.1 s for each. Please note that the time for prediction increases if the historical database increases, but this growth is linear in trajectory size and can easily be handled by optimizing the code and using parallelization.

3 RESULTS AND DISCUSSION

For the evaluation of the proposed algorithm, 105 trajectories collected in February 2024 were used. An overview over these trajectories as well as over the training data regarding the distribution of driving relation is shown in Fig. 5. These 105 trajectories were randomly shortened to arrive at partial trajectories (comp. Fig. 4). An overview over the length of the partial trajectories with regard to driving relation are shown in Fig. 6.

For all of the 105 partial trajectories a prediction (one single trajectory) was computed. The Average Displacement Error (ADE) and Final Displacement Error (FDE) of these predictions against the actually driven trajectories is shown in Fig. 7. ADE for the straight movements (ew, ew, ns, sn) is usually smaller than for the turning movements. This is rather intuitive, since first there are much more historical trajectories for the straight movements (comp. Fig. 5) and there is also much more variability in a turning movement in comparison to a straight movement. On the other hand, final displacement error shows a rather uniform distribution. This is also clear, since the movement ends at the egress lane and does not leave much room for deviation. The values for ADE and FDE reported here are in line with the values reported in (Wu et al., 2021), maximum ADE is less than 0.8 m and maximum FDE is even less than 0.25 m.

In a further analysis step, ADE vs. length of the partial trajectory was examined. Intuitively, one would expect ADE to decrease with increasing length of the partial trajectory. Looking at Fig. 8 certainly shows a trend, but this is much less pronounced than one might expect. For straight movements, Pearson's correlation coefficient is about -0.1, whereas it is -0.02 for turning movements. The disparity in available historical data could be one explanation for this effect.

In a last step, the numbers of candidates derived in the first step of the prediction algorithm vs. the length of the trajectory was analyzed. Again, one would suspect fewer candidates the longer the trajectory. As shown by Fig. 9, this is certainly true for movements entering the intersection from east or west along the two-lane streets. Still, two possible candidates are calculated most of the time. This is attributed to the inaccuracy in the initial data, which nearly equals the width of a lane, making it difficult to determine the lane the vehicle actually used in the past. On the other hand, vehicles entering from the north always get two candidates assigned (which is the maximum number of possible driving directions, a left turn is disallowed at this entrance to the intersection), whereas vehicles entering from the south always get assigned three candidates (also the maximum). This is attributed to the length of the partial trajectory, even at about 30 m into the intersection no driving relation can be ruled out.

4 CONCLUSIONS AND FUTURE WORK

In this paper, a simple and fast method to predict vehicle movement, based on historical V2X data, is proposed. Although no modern neural networks, like Long-Short-Term Memorys (LSTMs), are used, the results are comparable, especially for driving relations with sufficient historical data. As mentioned before, this is just a first step in an ongoing research effort and will act as a base line for further research.

In the current work, only the movement of vehicles is predicted. Especially at urban intersections, Vulnerable Road User (VRU) might have an influence on the motion of automated/autonomous vehicles. At the moment, there is no V2X message to track these traffic participants directly, although ETSI recently specified the VRU Awareness Message (VAM). There is also the chance of indirect detection through the use of Collective Perception Message (CPM), which allows vehicles and infrastructure to publish their sensor readings. Nonetheless, predicting VRUs also needs new approaches, as their possibility of movement is much more unconstrained than vehicular motion. Some possible methods for predicting pedestrian movement are already benchmarked in (Uhle-

[1] https://geopandas.org/en/stable/
[2] https://pypi.org/project/shapely/
[3] https://pypi.org/project/pyproj/
[4] https://python-visualization.github.io/folium/latest/

mann et al., 2023).

ACKNOWLEDGEMENTS

This research is financially supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) under grant number FKZ 19A22009F (VALISENS). We would like to thank Hsi Chen for carrying out some of the analysis steps.

REFERENCES

Banerjee, T., Huang, X., Chen, K., Rangarajan, A., and Ranka, S. (2020). Clustering object trajectories for intersection traffic analysis [clustering object trajectories for intersection traffic analysis]. *Proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems - VEHITS*.

Cosgun, A., Ma, L., Chiu, J., Huang, J., Demir, M., Añon, A. M., Lian, T., Tafish, H., and Al-Stouhi, S. (2017). Towards full automated drive in urban environments: A demonstration in gomentum station, california. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1811–1818.

ETSI EN 302 637-2 V1.4.1 (2019-04) (2019). ETSI EN 302 637-2 V1.4.1 (2019-04) Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. Standard, ETSI.

Hubmann, C., Schulz, J., Becker, M., Althoff, D., and Stiller, C. (2018). Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction. *IEEE Transactions on Intelligent Vehicles*, 3(1):5–17.

Klöppel-Gersdorf, M., Trauzettel, F., Koslowski, K., Peter, M., and Otto, T. (2021). The fraunhofer ccit smart intersection. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1797–1802. IEEE.

Kruskal, J. and Liberman, M. (1983). The symmetric time-warping problem: From continuous to discrete. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*.

Salvador, S. and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intell. Data Anal.*, 11(5):561–580.

Statistisches Bundesamt (Destatis), . (2022). *Verkehr Verkehrsunfälle 2021*. Statistisches Bundesamt (Destatis), 2022.

Uhlemann, N., Fent, F., and Lienkamp, M. (2023). Evaluating Pedestrian Trajectory Prediction Methods for the Application in Autonomous Driving. *arXiv e-prints*, page arXiv:2308.05194.

Wu, A., Banerjee, T., Rangarajan, A., and Ranka, S. (2021). Trajectory prediction via learning motion cluster patterns in curvilinear coordinates. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2200–2207.