Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 201 (2022) 150–157

Procedia Computer Science

www.elsevier.com/locate/procedia

The 13th International Conference on Ambient Systems, Networks and Technologies (ANT)
March 22 - 25, 2022, Porto, Portugal

Distributed Vehicular Communication Protocols for Autonomous Intersection Management

Hadi Jalali Lak*, Ashkan Gholamhosseinian, Jochen Seitz

Communication Networks Group, Technische Universität Ilmenau, Germany

Abstract

Intersections are considered to be a vital part of urban transportation and drivers are prone to make more mistakes, when driving through the intersections. A high percentage of the total fatal car accidents leading to injuries are reported within intersections annually. On the other side, there usually is traffic congestion at intersections during busy times of day. Stopping the vehicles in one direction to let the vehicle pass in the other directions leads to this phenomenon and it has a huge effect on traffic delay, which causes great squander in natural and human resources as well as leading to weather pollution in metropolises. The goal of this paper is to design and simulate different spatio-temporal-based algorithms for autonomous connected vehicles to be able to cross the intersection safely and efficiently. Vehicles employ vehicle-to-vehicle (V2V) communication via dedicated short range communications (DSRC) [4, 1] to exchange their kinematic information with each other. The proposed algorithms are compared to each other as well as with traditional methods like traffic lights in terms of various performance metrics such as traffic congestion, speed and especially delay to find the optimal control approach for autonomous intersection management.

Keywords: V2V; V2X; V2I; DSRC; TCL; TIC; FCFS; VIN.

1. Introduction

The most common way to control the traffic at intersections and roundabouts is using traffic lights or traffic signs [5]. With an increasing number of vehicles, size of cities and roads, these traditional control methods are getting less efficient and safe. On the other side, traditional traffic control methods introduce extra delays and increase trip time,

* Corresponding author. Tel.: +49-177-8699043
E-mail address: hadi.jalali-lak@tu-ilmenau.de

1877-0509
10.1016/j.procs.2022.03.022

which leads to a huge waste in time and resources. According to a report published by Texas Transportation Institute, every commuter suffers from average delay of 34 hours per year which costs more than 100 billion dollars [6].

Autonomous driving is one of the technologies which has grown rapidly in the recent years and is still progressing fast. Nowadays, we see that a lot of companies and research centres are working hard on improving autonomous vehicles. Although there are some discussions about safety and reliability of autonomous driving, it seems that this technology will play a very important role in future traffic strategies. So, all traffic rules and infrastructures may be different to be more compatible with the new generation of cars. One fundamental technology for future automobile assistance systems is vehicle-to-everything (V2X) communications including vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. This technology will help developers of such systems provide more reliable tools by letting them exchange critical information like warnings, recommendations, and status, which a vehicle cannot access by itself.

The project described in this paper is based on [1, 2, 3]. A short range vehicular network is used to transfer the data. Moreover, communication is only vehicle to vehicle since there is not any third party element or road-side unit (RSU). All vehicles approaching the desired junction are using DSRC to send their information such as current location, speed, planned route, etc. Messages will be sent in a specific period and all the messages should be received and processed by all the other vehicles at the intersection area. When a vehicle checks all the received information and compares them, it will decide how to behave at the intersection. However, some parameters are very important from the aspect of safety. For instance, an inaccurate position information could lead to a collision between two or more vehicles.

In this study, we rely on the measuring tools like GPS of each vehicle and ignore probable errors. We have used the simulation tools “Simulation of Urban MObility” (SUMO) [8], the “Objective Modular Network Testbed in C++” (OMNeT++) [7] and “Vehicles in network simulation” (Veins) [9] to design, implement, simulate and animate our network and related controlling algorithms.

The remainder of the work is organized as follows; Section 2 includes the basic definitions and rules which should be applied to all vehicles. Section 3 gives an overview of protocols and algorithms used in the controlling models. Section 4 shows final simulation results and section 5 concludes this work along with suggestions for future work.

2. Assumptions and Rules

Firstly, we describe the conditions needed to be met for two or more vehicles to collide at the intersection. We assume that the intersection is a square area which can be divided into smaller square boxes. Entry and exit points are predefined as a constant distance from the real entrance and exit points of the intersection. The intersection has two entering lanes from all directions and is discretized by the cells, each characterized with a unique identifier [2].

2.1. Collision Detection

Every vehicle should be able to update its trajectory cells list (TCL) when it is crossing the intersection. Therefore, it is necessary for all the vehicles to know their own location and the occupied cell number inside the grid of the intersection. To achieve this goal, all vehicles are equipped with a GPS device. Besides, they have access to the digital map database of the intersection’s coordinates. It is the proper way to update the TCL and share the updated information with other vehicles in the desired range of the intersection. If a vehicle is still at the beginning of the intersection grid, it will not alter its TCL and the list will show the full set of the cells that will be used for crossing the intersection. But, if the vehicle is inside the junction grid, it should use the current block number to modify the list as follows. The current cell number is known and the TCL is a sorted list. So, all the cells in the list which are located before the current one should be removed from the trajectory list. This procedure is visualized in figure 1.

If two or several vehicles have space and time conflict at the intersection area, it means that there is a collision. In other words, there is a collision probability between two vehicles if they have overlapping time intervals (arrive and exit time) and they have at least one common cell in their TCL while passing the intersection area. Otherwise, they will not collide and both can pass the intersection safely [2].

In our project, the collision detection algorithm for intersections should run on all vehicles. Information for this purpose will be gained from the safety massages broadcast by surrounding vehicles. If the algorithm finds one or more

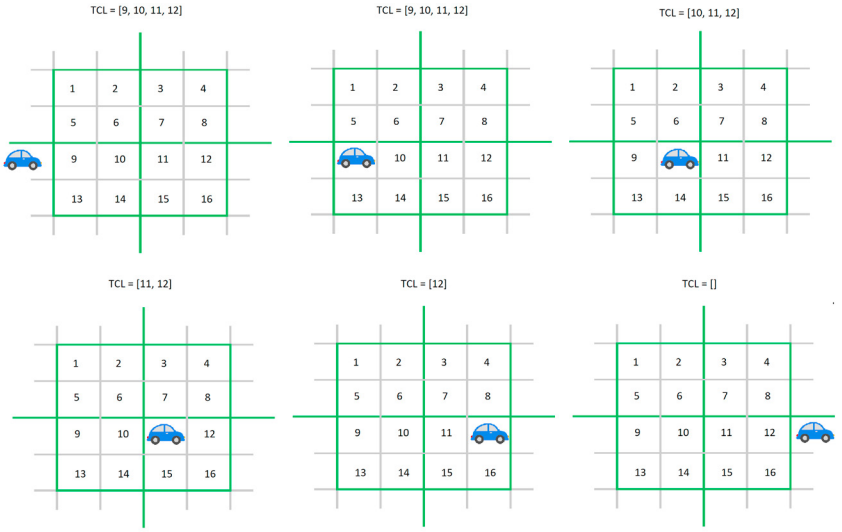
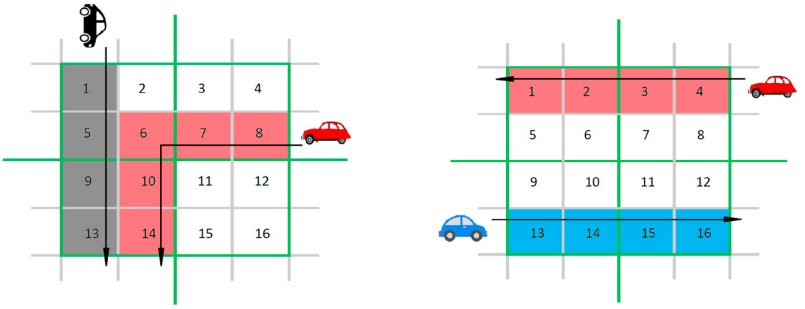


Fig. 1: TCL update

common cells between the TCL of another vehicle with its own TCL, it will return number of the cell which causes the first conflict. This cell is named as trajectory intersecting cell (TIC) [2]. Figures 2a and 2b show two scenarios in which the vehicles do not have any TIC and they can pass the intersection without stopping. In figures 3a and 3b, two other situations are shown in which two vehicles have common cells in their planed trajectory list. So, a collision could happen between these cars if they want to cross the intersection at the same time.



(a) (b)

Fig. 2: Passing without conflict

If collision detection algorithm detects any possible conflict, First Come First Served (FCFS) will be used to resolve the conflict. This algorithm assigns the higher priority to the vehicle which has the lower arrival time and gets the entrance point of the junction earlier. So, this vehicle can cross the intersection before the others with greater arrival times. If two vehicles arrive at the intersection at the same time, priority will be assigned according to the road importance. It means that every vehicle that is driving on the primary road has the priority to another one on the secondary road. If the vehicles have equal conditions, the priority policy will let the vehicle with higher vehicle identification number (VIN) pass the intersection before the other one. VIN is an unique number assigned to each vehicle [2].

2.2. Vehicle status

We have defined four states for the vehicle considering its location relative to the desired intersection.

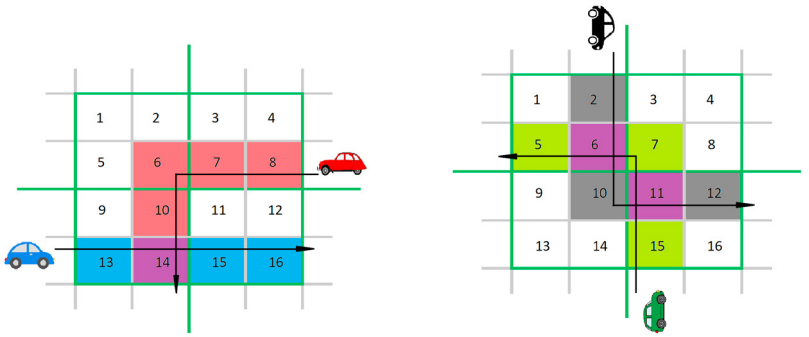


Fig. 3: Conflict in TCL

- Approach: The vehicle's distance to the desired junction is less than a predefined specific parameter.
- Wait: The vehicle is stopped at the entrance of the junction and waits for other vehicles with higher priority to cross.
- Enter: The vehicle is inside the junction grid and tries to pass the intersection.
- Exit: The vehicle has already left the junction area, its distance from the intersection is more than a specific predefined threshold.

2.3. Intersection Safety Messages

In our protocols, every vehicle is allowed to use three types of intersection safety messages to communicate with the surrounding vehicles in the range of the intersection [3].

- The *ENTER* message informs the surrounding vehicles that the vehicle is approaching the intersection area with a specific crossing route. It contains eight parameters: vehicle id, current road segment, current lane, next road segment, arrival time, exit time, TCL, cells arrival time, message sequence number and message type, which is *ENTER* in this case.
- The *CROSS* message informs the surrounding vehicles that the vehicle has already arrived inside the junction crossing area. This message consists of the sender's identification and trajectory details, identifying the space that will be occupied by the vehicle while crossing the intersection. The *CROSS* message contains the same parameters as the *ENTER* message. Its trajectory cells list contains the updated list of trajectory cells, their related arrival and exit times for the current cell, remaining cells along the vehicle's trajectory through the intersection area, and the message type which is *CROSS* in this case.
- The *EXIT* message shows that the vehicle has exited the intersection region. The *EXIT* message carries three parameters: vehicle id, message sequence number, and the *EXIT* message type.

3. Intersection Management Protocols

3.1. Concurrent Crossing-Intersection Protocol (CCIP) [2]

In this method, the conflicting vehicle with higher priority can ignore the intersection safety messages from the vehicles with lower priority and cross the intersection without slowing down or stopping. Furthermore, vehicles which have lower priority are supposed to completely stop in the entrance of the intersection if they cannot win the competition against another vehicle. They should wait until they receive the *EXIT* message from the high priority vehicle to be able to cross the intersection. This protocol is applied across all priority levels. The algorithms 1 and 2 represent the behaviour of sender and receiver vehicles respectively.

Algorithm 1 Send status

```
Input: state of the vehicle
Output: intersection safety message
if STATE = Approach or STATE = Wait then
    Broadcast ENTER message
else if STATE = Enter then
    Broadcast CROSS message
else if STATE = Exit then
    Broadcast EXIT message
end if
```

Algorithm 2 CCIP on receive behaviour

```
Input: Received safety message from another vehicle M
Output: Blocking vehicles list BL
if M = CROSS then
    check the trajectory lists and find the conflicts, TIC
    if TIC != empty then
        add vehicle's Id to BL
    end if
else if M = ENTER then
    run priority assignment algorithm
    if other vehicle has higher priority then
        add vehicle's Id to BL
    end if
else if M = EXIT then
    if vehicles Id is in BL then
        remove the Id from BL
    end if
end if
```

3.2. Maximum Progression-Intersection Protocol (MPIP) [2]

This method is similar to the previous one. So, the higher priority vehicle will cross the intersection ignoring the others with lower priorities. But the lower priority vehicle is also allowed to enter the intersection boundaries until getting in front of the conflicting cell. Here, it should come to a complete stop and wait for a message from the high priority vehicle indicating that is has passed the corresponding cell and there is no conflict anymore. Now, the lower priority vehicle can continue to pass the intersection. Similar to CCIP, the algorithm 1 applies to the sending vehicle, but the receiving vehicle should act based on algorithm 3.

Algorithm 3 MPIP on receive behavior

```
Input: Received safety message from another vehicle M
Output: TIC
if M = CROSS or M = ENTER then
    check the trajectory list and find the first conflicting cell CC
    if CC != NULL then
        run priority assignment algorithm
        if other vehicle has higher priority and CC has lower index in vehicle's own TCL rather than the current TIC then
            TIC = CC
            save sender's Id
        end if
    else
        run priority assignment algorithm
        if other vehicle has higher priority and it is the same sender saved in the memory then
            remove corresponding TIC
            remove sender's id from memory
        end if
    end if
else if M = EXIT then
    if vehicle's Id is saved in the memory then
        remove corresponding TIC
        remove sender's Id from memory
    end if
end if
```

3.3. Spatio-Temporal Intersection Protocol (STIP) [3]

The aim of this method is to increase the parallelism inside the intersection area by letting more vehicles cross the intersection simultaneously. This goal is achieved by allowing even conflicting vehicles to make maximal progress inside the intersection area without sacrificing the primary goal of safety. In this protocol, vehicles will take the temporal information regarding arrival and exit times gained from the other vehicles in the vicinity more into account and make smarter decisions. If the lower priority vehicle detects a conflict with a higher priority one, it will check if arrival and exit times of the other vehicle to the common cell overlap with arrival and exit times of its own. If yes, it will stop before entering the conflicting cell and wait until it is cleared. Otherwise, it will continue to pass the intersection. Algorithm 4 determines the movements of the receiver vehicle. The sender vehicle uses algorithm 1.

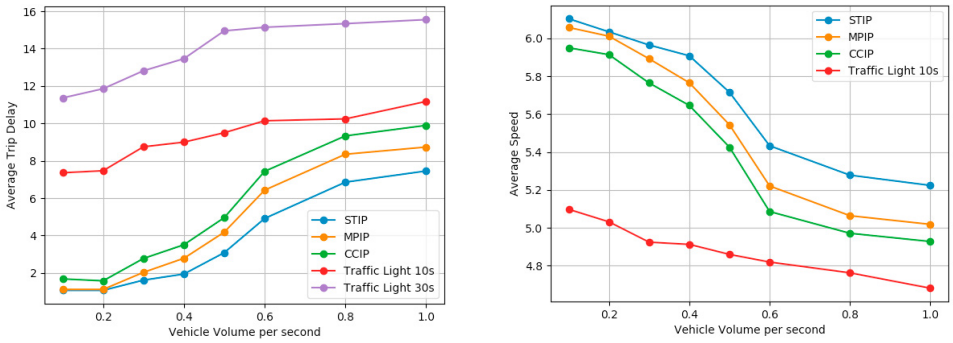
Algorithm 4 STIP on receive behaviour

```
1:  Input: Received safety message from another vehicle M
2:  Output: trajectory intersecting cell (TIC)
3:  if M = CROSS or M = ENTER then
4:      check the trajectory list and find the first conflicting cell CC
5:      if CC != NULL then
6:          run priority assignment algorithm
7:          if other vehicle has higher priority and CC has lower index in vehicle's own TCL rather than the current TIC then
8:              check arrive and exit time of sender to the conflicting cell
9:              if there is overlap with vehicle's own arrive and exit times then
10:                 TIC = CC
11:                 save sender's Id
12:             else
13:                 if vehicle's Id is saved in the memory and TIC = CC then
14:                     remove corresponding TIC
15:                     remove sender's id from memory
16:                 end if
17:             end if
18:         end if
19:     else
20:         run priority assignment algorithm
21:         if other vehicle has higher priority and it is the same sender saved in the memory then
22:             remove corresponding TIC
23:             remove sender's id from memory
24:         end if
25:     end if
26: else if M = EXIT then
27:     if vehicle's Id is saved in the memory then
28:         remove corresponding TIC
29:         remove sender's Id from memory
30:     end if
31: end if
```

4. Evaluation

A road map area of $200m \times 200m$ with no obstacles has been designed in SUMO. It is a 4-way intersection. Trip time is defined as the time in which a vehicle travels from the start to the end point. In our case, start point is 100m before the intersection and end point is 100 meter after the intersection. Trip time is calculated by SUMO for each vehicle considering the vehicle's parameters as maximum speed and acceleration. It is assumed that the vehicle will not come to any stop during its trip. At the end of the trip, real travel time is measured and compared to the estimated trip time. The difference between the above-mentioned values is known as trip delay. This delay is caused by the stops occurring while passing the intersection. The average delay across all the vehicles involved in the simulation is taken as the metric.

Figure 4a for symmetric traffic demonstrates the comparison of average trip delay among the protocols to the traditional traffic light models, one programmed to switch between red and green lights every 30s and the other every 10s. Yellow light duration in both of them is 3s. Traffic volume is defined as cars per second for the whole intersection, not for each leg of it. Vehicle instances are not created at the same time. They are introduced to the simulation one by one in the row. Apparently, our intersection management models have significantly higher performance in comparison to the traditional ones. We can see that CCIP has improved the delay time and throughput of the intersection by 62.74% and 44.1% in comparison to 10s and 30s traffic light systems respectively. MPIP is better than 30s traffic



(a) Delay for symmetric traffic

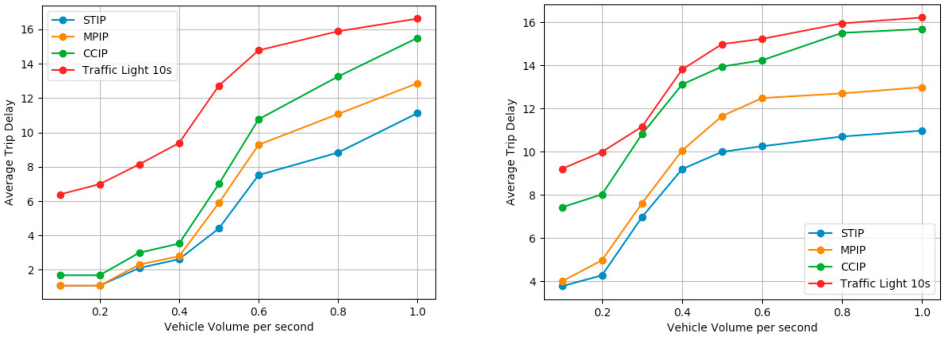
(b) Speed for symmetric traffic

Fig. 4: Simulation results for symmetric traffic

light, 10s traffic light and CCIP by 68.6%, 52.8% and 15.58%. Finally, STIP is the best working algorithm and has the lowest delay time of all alternatives. Its functionality is 74.65%, 61.96%, 31.95% and 19.41% better than 30s traffic light, 10s traffic light, CCIP and MPIP. For low traffic volumes, the trip delay is low and negligible in all the three management models. As the traffic volume increases, the average delay of the CCIP model also increases and after the traffic volume reaches 0.8, it gets near the performance of the traffic light model 10s, but is still slightly better. In the case of MPIP, the average delay has a low value up to traffic volume of 0.5. After that point, it starts to increase, but the values are always under the corresponding values of traffic light models and CCIP. The STIP model performs with a significantly small average delay for the traffic volumes under 0.7 while beyond that, it gets higher and reaches its maximum at the end of the simulation which is around 11s. We have calculated the average speed of the vehicles for every simulation. The result is shown in figure 4b. As expected, the highest average speed belongs to the vehicles involved in the STIP simulation. It is around 6.1 meter per second for traffic volume of 0.1 vehicle per second, and decreases to 2.3 at the end for the denser traffic volume of 1 vehicle per second. The other algorithms end up to a lower average speed, but still all have better results in comparison to the traffic light model. We will ignore the average speed graph in the next tests because its behaviour is predictable from the average delay graph.

Next, we studied another test scenario in which a higher traffic is applied to two legs of the intersection. The other legs remain the same as the previous scenario. The lanes with the higher number of cars belong to the primary road. All the models behave almost equally to the previous simulation scenario below traffic volume of 0.4. After this point, the average delay grows faster and reaches to a higher value at the end. In the case of CCIP, the value is almost the same as the traffic light model at the end, but the other two models are still performing better in comparison to the traffic light model. Figure 5a depicts the results of this experiment. We ignored the 30s traffic light model because of its weak results. In this case, also we can obviously see that for the traffic volumes below 0.4 vehicles per second, the average delay is very low for CCIP, MPIP and STIP models. Above this point, because of the higher traffic density of the priority road, some kind of saturation starts to happen and the average delay value grows with a higher rate. But at the end, the traffic light model still performs worse than the other models. We can see that CCIP has improved the throughput of the intersection by 38.01% in comparison to 10s traffic light model. MPIP is better than the 10s traffic light and CCIP by 49.01% and 17.9%. As expected, STIP is the best working algorithm and has the lowest delay time of all. Its functionality is 57.4%, 31.35% and 16.4% better than 10s traffic light, CCIP and MPIP.

In the next scenario, we increased the traffic volume. This time, the X-axis is representing the traffic volume per each leg of the intersection and not the whole intersection. It means that we have a four times higher traffic rate than in the first scenario. Here, we make the situation more difficult and create vehicle instances simultaneously each time in all the legs. So, they reach the intersection crossing area in very close times and passing the intersection would be more challenging. According to figure 5b, it can be recognized that CCIP is working better than the traffic light model only for the traffic volumes under 0.3 vehicles per second, but after that these two models have almost the same performance. MPIP is better than 10s traffic light model and CCIP by 28.3% and 22.6%. STIP is again the best working algorithm and has the lowest delay time of all. Its functionality is 38.1%, 33.41% and 13.5% better than 10s traffic light, CCIP and MPIP models.



(a) Delay for asymmetric traffic

(b) Delay for per leg traffic

Fig. 5: Simulation results for asymmetric traffic

5. Conclusion and Future Work

In this paper, we have implemented and simulated three DSRC-based protocols for crossing the intersections safely and without collisions in the most efficient way. We tried to increase the parallel movement inside the intersection grid and not sacrificing the safety standards at the same time. We achieved a better throughput for the intersection and lower trip delay using three V2V-based communication protocols in comparison to the traditional traffic management method. The suggested controlling models and protocols have still room for improvement.

Following are some aspects that can be considered in future works to achieve a better functionality and performance; dealing with some special events which might happen during crossing the intersection (for instance, accidents, a defective vehicle, etc.), extension of the intersection grid, utilization of vehicles with different shapes and sizes, integration of vehicles with special priorities like emergency vehicles, smart selection of speed and acceleration by the vehicles instead of coming to a complete stop and wait, investigation of some possible errors and problems that may happen while exchanging the security messages (for example, GPS inaccuracy, packet loss, etc.), and implementation of the protocols for other types of intersections such as roundabouts.

References

[1] Seyed Azimi, Gaurav Bhatia, Ragunathan Raj Rajkumar, and Priyantha Mudalige (2011) "Vehicular Networks for Collision Avoidance at Intersections." *SAE International Journal of Passenger Cars – Electronic and Electrical Systems*, **4**, pp. 406–416.

[2] Seyed Azimi, Guarav Bhatia, Ragunathan Raj Rajkumar, and Priyantha Mudalige (2013) "Reliable Intersection Protocols Using Vehicular Networks." *2013 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, Philadelphia, PA, USA, pp. 1–10.

[3] Seyed Azimi, Guarav Bhatia, Ragunathan Raj Rajkumar, and Priyantha Mudalige (2014) "STIP: Spatio-Temporal Intersection Protocols for Autonomous Vehicles." *2014 ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*, Berlin, Germany, pp. 1–12.

[4] John B. Kenney (2011) "Dedicated Short-Range Communications (DSRC) Standards in the United States." *Proceedings of the IEEE*, **7**, pp. 1162—1182.

[5] Michigan Department of Transportation (MDOT) (2021) "What is a Roundabout?" http://www.michigan.gov/roundabouts.

[6] Texas A&M Transportation Institute (2021) "The 2021 Urban Mobility Report." http://mobility.tamu.edu/ums/.

[7] András Varga and OpenSim Ltd. (2016) "OMNeT++ Version 5.6.1." https://doc.omnetpp.org/omnetpp/SimulationManual.pdf.

[8] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner (2018) "Microscopic Traffic Simulation using SUMO." *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, USA, pp. 2575–2582.

[9] Christoph Sommer, David Eckhoff, Alexander Brummer, Dominik S. Buse, Florian Hagenauer, Stefan Joerer, and Michele Segata, Michele (2019) "Veins – the Open Source Vehicular Network Simulation Framework." In A. Virdis and M. Kirsche (eds.): *Recent Advances in Network Simulation*. EAI/Springer Innovations in Communication and Computing. Springer, Cham.