A Realistic Simulator for the Design and Evaluation of Intelligent Vehicles

Sven Gowal, Yizhen Zhang and Alcherio Martinoli

Abstract—**The number of vehicles hitting the road each day is rapidly increasing, and several problems, such as traffic congestion or driver safety, can no longer be solved in the same fashion as before. Intelligent transportation systems could potentially solve part of these problems, but prototyping, designing and testing cooperative smart vehicles is a cumbersome task. This paper presents a realistic simulator where intelligent vehicles can be designed and analyzed with a pragmatic approach. A number of advances in robotics have already been transferred to vehicular technology, with a potential increase of this trend into the future. Here, we develop a plugin for a well-established robotics simulator (Webots), in order to reinforce at the virtual level this cross-fertilization between the two areas and create a baseline for realistic studies of future solutions in real intelligent vehicles.**

I. Introduction

The need to create intelligent vehicles that can adapt to the current traffic context and possibly to the individual driver behavior becomes more important every day. Intelligent transportation systems consisting of hundreds of intelligent vehicles which sense, decide, and act in a distributed fashion in the same shared environment can be designed and controlled in different ways. It is important that models and algorithms can be assessed in a realistic manner. Unfortunately, deploying multiple prototype vehicles in reality is very difficult, because of cost and safety issues, and it is thus necessary to simulate these complex systems. Simulation can help in the design, optimization and performance assessment of intelligent vehicles before their deployment in reality. This includes systematic validation of the positions of their different sensors [25] as well as the underlying actuation procedures. The closer the simulation is to reality, the easier and faster the transition to reality becomes.

Several types of vehicular simulations exist, ranging from macroscopic to microscopic, characterized by more or less realism. Macroscopic traffic simulations, also called continuous flow simulations, are mainly used in traffic flow analysis and capture average behaviors. Examples of such simulations are the British TRANSYT-program and the American FREQ and FREFLO-programs [1, 2]. This type of simulation is out of scope for our objectives because it cannot capture individual behaviors and thus the analysis of a single intelligent vehicle in standard traffic is not possible.

The general trend of today's traffic analysis lies in microscopic simulations. Microscopic simulations such as Sumo [3] can capture individual behaviors and can be extended to analyze car-to-car communication [4]. Today's most widely used microscopic simulators are VISSIM [6], PARAMICS [7], CORSIM [8] and AIMSUN [9]. They have been calibrated to match actual traffic flows and can handle several thousands of vehicles running in a complex road network that includes traffic lights and even pedestrians (for VISSIM). Even though these simulators differ on certain aspects, they all use simple kinematic models derived from the Newtonian dynamics rather than dynamic models. Additionally, complex maneuvering such as lane changing or parking is discretized. Our interest is to develop simulation tools that can assess not only traffic flow but also local maneuvering such as swerving to avoid an obstacle partially blocking a lane. Thanks to the realism of these tools, the design and evaluation of vehicles endowed with sophisticated driving assistance modules [26] or the development of fully autonomous vehicles that use complex sensory apparatus (such as the vehicles that took part in the DARPA Urban Challenge 2007 [10]) is also possible. This paper should therefore be considered as a complementary effort to the aforementioned simulation tools.

Although simulators which exhibit realistic dynamics for a single vehicle exist, such as CarSim [5] and SiVIC [11], they do not offer the possibility to assess a potentially intelligent vehicle inside an actual traffic scenario with realistic sensing capabilities. In this work, we propose a realistic simulator extension, which we incorporate here as a plugin for Webots [12], a commercially available robotics simulator that we will describe in Section II. This plugin does not only capture the realistic physical dynamics of each vehicle, but also enables us to control each vehicle individually (i.e. with different controllers) when necessary. We can potentially analyze real traffic scenarios exhibiting specific behaviors for each car. Vehicles can sense the environment with realistic discrete sensors (reproduced with calibrated noise and nonlinearities and available through Webots' sensors library) and sensor fusion algorithms can be tested before their deployment in reality. Additionally, vehicles can communicate in a realistic fashion with an OMNeT++ [13] plugin [14]. With all the above mentioned elements, algorithm development for multi-vehicle environments such as the one presented in [15, 16] can be accelerated.

II. Towards Simulating Intelligent Vehicles in Realistic Traffic Scenarios

In this section, we describe the main parts of our simulation engine.

Sven Gowal and Alcherio Martinoli are with the Distributed Intelligent Systems and Algorithms Laboratory of the Ecole Polytechnique Fédérale de Lausanne. `svenadrian.gowal@epfl.ch`, `alcherio.martinoli@epfl.ch`

Yizhen Zhang now works for Applied Materials as a mechanical engineer.

A. Webots

Webots is a mobile robotics simulation software that provides a rapid prototyping environment for modeling, programming and simulating mobile robots. Webots 6 uses the Open Dynamics Engine (ODE) [18] library for realistic physics simulation. The ODE library is an open source, high performance library for simulating rigid body dynamics. Customized physics and vehicle dynamics properties can be implemented in Webots based on ODE. Therefore a real car-based vehicle model with realistic vehicle dynamics features can be developed in Webots with ODE. Webots can also be extended with a realistic communication model. In particular, a realistic radio communication between robotic agents was implemented in [14] by wrapping the OMNeT++ network simulation engine, as a plug-in for the Webots simulator. OMNeT++ is a public-source, component-based, modular and open architecture for discrete event simulation quite suited for wireless communication networks. Overall, the latest version of Webots 6 provides a useful platform for developing a dynamic embodied simulation of multiple, intelligent vehicles.

B. Road Network

The road network and its properties are directly extracted from the OpenStreetMap (OSM) XML data [17]. The exported maps are translated into an internal graph data structure that can be used by the intelligent vehicles if needed. This structure can potentially be analyzed for shorter, safer or less congested paths. Figure 1 shows how an actual map of the region near EPFL, Lausanne, Switzerland has been imported in our simulation environment.

OSM XML data offers a complete set of different types of road segments (called ways). Each way is composed of a set of geographic locations (called nodes) and properties. The main properties used when converting OSM data into a usable road network for our traffic simulator are the following: the segment type (ranging from motorways to residential roads), the direction of the road (one-way versus both-ways) and the number of lanes. These properties are analyzed to generate the internal segment lanes and connect the lanes together at intersections. Unfortunately, to automate this process, we need to make some assumptions about the road network. For example, a motorway link branching off a motorway segment will automatically be connected to the rightmost lane of the motorway (this would be true in most situations). Without being exhaustive, and as an example, below are a few types of connections that are handled specially:

- Segments branching off multi-lanes roads,
- Splitting segments,
- Multi-lanes to single lane segment or
- Single-lane to multi-lanes segment.

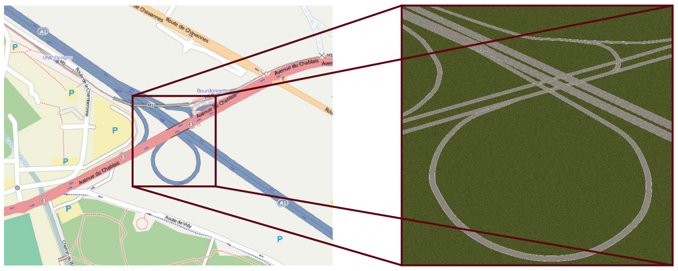

Fig. 1. Conversion from the OpenStreetMap data to a usable simulation world.

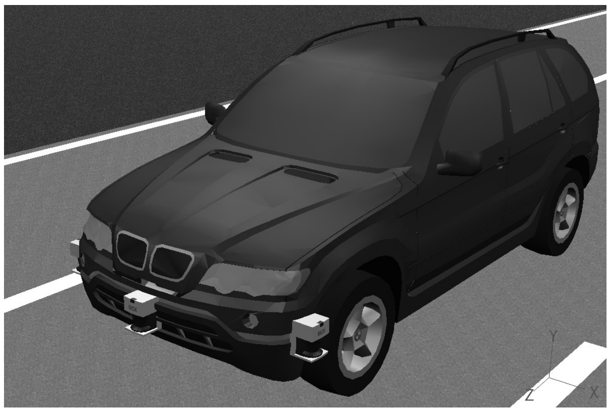

Fig. 2. Screen Shot of a dynamic embodied vehicle model. This vehicle is equipped with four SICK LMS 291 laser rangefinders.

It is important to note that a variety of OSM editors are available and can be used to create custom maps. Other GIS file formats such as *Shapefile* can easily be converted to OSM XML data if needed.

C. Car Model

The model used in our simulator and shown in Figure 2 is composed of the vehicle body and four wheels. The vehicle body is able to move freely in all six DOF in the 3D space, the wheels can all spin and move vertically relative to the body, and the steering wheels can also yaw. So the whole vehicle model has 16 DOF in total ($16 = 6 + 3 \times 2 + 2 \times 2$), i.e., six DOF for the vehicle body, three DOF for each of the two steering wheels, and two DOF for each of the two non-steering wheels. Built in Webots based on ODE, this model already incorporates basic rigid dynamics properties including typical steering dynamics response. In the rest of this section, we describe the models that play an important role when modeling a car as well as a traffic system.

1) Joint Model: Wheels are linked to the vehicle body by the ODE *hinge-2* joint, which is defined especially for car simulations. The hinge-2 joint is equivalent to two hinges connected in series, with different hinge axes. An example is the steering wheel of a car, which can both spin and steer along different axes. Therefore all three motions (wheel steering, spinning, and vertical movements) of a steering/driving wheel can be conveniently integrated into just one joint model. For consistency, four hinge-2 joints can be applied between the vehicle body and its four wheels, respectively, where the steering motions of the rear (non-steering) wheels can be simply turned off. This is a more compact and integrated joint model especially customized for dynamic vehicle simulations.

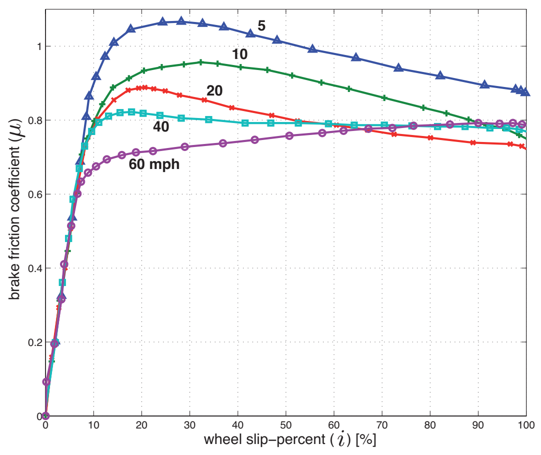

Fig. 3. Variation of the friction coefficient μ on dry asphalt [19].

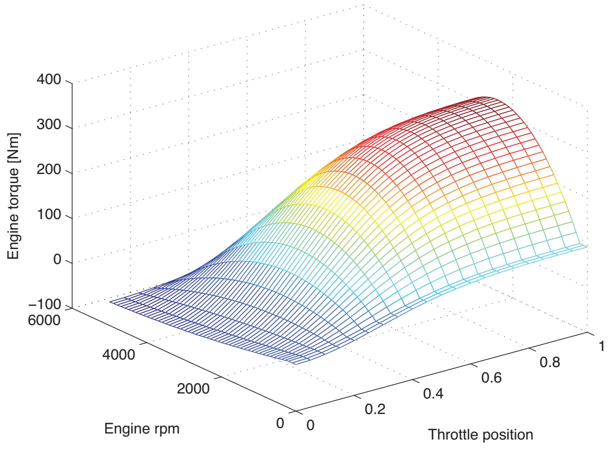

Fig. 4. Example of throttle to torque conversion.

2) Friction Model: The friction coefficient μ between the tire and the road is an important parameter limiting the maximum tractive and braking forces generated from the tire-road contact, and it varies under different environmental and dynamical conditions. Figure 3 shows how μ varies for different speeds on dry asphalt. We can define the longitudinal slip as:

$$i = \begin{cases} \left(1 - \frac{v}{r\omega}\right) & \text{if } v \leq r\omega, \text{ tractive slip} \\ \left(1 - \frac{r\omega}{v}\right) & \text{if } v \geq r\omega, \text{ braking slip} \end{cases} \tag{1}$$

where v is the linear speed of the wheel center, ω its angular speed and r its radius. The original, oversimplified ODE friction model leaves μ equal to 1. Although the Pacejka "magical" model [20] is a better approximation, a simpler and more general model was desired and we decided to only slightly improve the ODE friction model with the equation below.

$$\mu = \begin{cases} \mu_1 & \text{if } i \leq 20\% \\ \mu_2 & \text{otherwise} \end{cases} \tag{2}$$

Setting μ_1 to 1.0 and μ_2 to 0.8, this equation captures the main characteristics of the curve in Figure 3 and avoids complex nonlinear mathematical equations with many model parameters to be tuned with real experimental data. Of course, μ_1 and μ_2 used can potentially be modified to account for different road conditions.

3) Kotwicki's Engine and Brake Model: Standard equations governing the engine are described in [21]. In our simulator, we are only interested in having a throttle position to engine torque converter. To that extent the following formula is proposed:

$$t_e = t_{\text{eff}}(\tau) \cdot T_f(w_e) + (1 - t_{\text{eff}}(\tau)) \cdot T_d(w_e) \tag{3}$$

where t_e is the engine torque, τ the throttle position ($\tau \in [0, 1]$), t_{eff} is the non-linear mapping function between the throttle position and its actual effect, w_e is the engine's angular velocity, T_f is the burning torque mapping for a particular engine regime and T_d is the friction torque mapping for a particular engine regime. T_f and T_e are parameterized as second order equations and can be tuned with real engine specifications (mean square error fitting). The effective throttle mapping function t_{eff} represents how the gas pedal influences the combustion and thus the developed torque. Finally the effective throttle mapping function is estimated by:

$$t_{\text{eff}}(\tau) = 1 - e^{-a\tau^b} \tag{4}$$

with a and b both positive, where a and b would be proportional to the start (when the throttle position is close to 0) and end (when the throttle position is close to 1) slack of the pedal respectively. Figure 4 shows an example of throttle and engine rpm to torque conversion.

In a similar way, the braking action is proportional to the effective brake applied:

$$t_b = b_{\text{eff}}(\beta) \cdot b_{\text{max}} \tag{5}$$

where t_b is the braking torque, b_{eff} is the mapping between the brake pedal and its actual effect and b_{max} is the maximal braking torque. The brake effects are simulated by applying a counter torque on the wheels.

4) Sensor Models: Any of the sensors available in the Webots sensor library can be used and added to each car independently. Each of these sensors is reproduced with calibrated noise and nonlinearities. Specific sensors include cameras, compasses, GPS, gyroscopes and distance sensors. Additionally, to make the simulation time lower, our plugin can provide (if necessary) information that would be difficult to compute from real sensor data (such as the position of other vehicles).

D. Driver Model: Modified Helbing Model

A great advantage behind our simulator is that each vehicle can potentially be controlled by any type of controller. In other words, the decoupling of the driver model from the driver assistance (and the underlying intelligence) is possible and different driver models can be tested with different vehicle setups. Meanwhile, to generate standard traffic, we provide a simple yet realistic driver model, namely a modified Helbing Model. This model is based on Helbing's intelligent-driver model (IDM) [22] for car-following. The key point of car-following behavior is setting vehicle acceleration according to the current situation and driver preferences, which include the current gap and desired gap between the host vehicle and the lead vehicle, current speed and relative speed to the lead vehicle, driver's preferred speed and driving style. Helbing presented the formula:

$$a_x = a_{\text{acc}} \left[1 - \left(\frac{v}{v_{\text{pref}}} \right)^{\alpha} - \left(\frac{R^*(v, RR)}{R} \right)^2 \right] \quad (6)$$

to compute the desired acceleration a_x, where a_{acc} is the maximum acceleration limit, v and v_{pref} are the current and preferred host vehicle speed respectively, α is the acceleration exponent parameter and R and R^* are the current and desired gap between the host vehicle and the lead vehicle respectively. The desired range R^* dynamically varies with current host vehicle speed v and the closing speed relative to the lead vehicle (range rate RR):

$$R^* = R_0 + R_1 \sqrt{\frac{v}{v_{\text{pref}}}} + v \cdot t_h - \frac{v \cdot RR}{2\sqrt{a_{\text{acc}} \cdot a_{\text{pref}}}} \quad (7)$$

where R_0 and R_1 are constant distance parameters, t_h is the preferred time headway and a_{pref} is the preferred deceleration. When navigating on a lane, the host vehicle creates a trajectory described by northing and easting coordinates (gathered from the map generated when reading the OSM XML data in Section II-B) as well as velocity and acceleration information. This trajectory is analyzed and followed with the aid of the proportional-integral (PI) controller from Linderoth et al. [23], which actuates the steering as well as the throttle and brake pedals.

On top of Helbing's car-following model, a lane change behavior is added. Closely related to the *projected minimum distance* which is a quantitative measure characterizing the emergency level of rear-end collisions [26], the lane change decision is calculated using the prevailing range and vehicle speeds. The assumption is that the lead vehicle will brake at a constant maximum deceleration level (a_{max}) until it comes to a stop, while the host vehicle decelerates at its preferred deceleration a_{pref}. If the projected position of the two vehicles comes within a certain range R_{thres}, the host vehicle will change its current lane if possible (i.e. collisions are also not possible with cars on the future lane). So, when:

$$\frac{(v + RR)^2}{2 \cdot a_{\text{max}}} - \frac{v^2}{2 \cdot a_{\text{pref}}} + R < R_{\text{thres}} \quad (8)$$

the host vehicle will change its lane. Collision checks for the future lane are done in the same manner. On the other hand, if the host vehicle cruises for a period of time t_f at a fraction v_{thres} of its preferred speed behind the lead vehicle, it will also try to change its lane. When changing its lane, the host vehicle creates a maneuver reaching its desired new lane. The resulting trajectory is then followed with the aid of the PI controller.

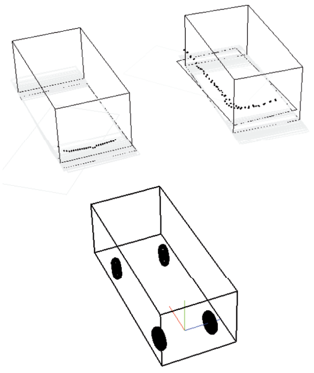
Fig. 5. Screen shot of the debug window of our intelligent vehicle. The points show the raw data returned by the SICK LMS 291 sensors. The two boxes are the other vehicles estimated position.

E. Traffic Model

1) Kinematic Model: Our plugin is able to realistically simulate several tens of vehicles at a speed higher than real-time on a traditional Intel® Core$^{\text{TM}}$2 Duo running at 3 GHz with 4 GB of RAM (i.e. 20 realistic vehicles run at 3.5 $\times$ real-time). If appropriate, the simulation speed can be further increased by using simple kinematic models whilst keeping the other features unchanged. This avoids the extra computation of solid dynamics for cars that do not need that level of realism.

2) Rail-Based Model : The plugin can also, similar to standard microscopic simulators, generate a complete traffic scenario including up to a few thousand vehicles. These vehicles are driven along virtual rails and only control their desired lane and longitudinal acceleration using the driver model presented in Section II-D. Their direction at intersections is randomized according to predefined ratios (if provided). This feature creates a potentially hybrid traffic: rail-based vehicles can be added and run in parallel with other more realistically driven cars, embedding different sensing and actuation mechanisms.

III. Show-case Scenario

In this section, we will present three simple scenarios to show-case how sensors and communication can be tested by extracting potentially useful information from our simulator. The experiments shown here provide an insight on the capabilities of our plugin and do not focus on the details of our underlying algorithms.

A. Our Intelligent Vehicle

Visible on Figure 2, we have equiped our intelligent vehicle with four simulated SICK LMS 291 sensors so as to cover a 360° field of view. The SICK LMS 291 is a laser rangefinder, which scans at 75 Hz over 180° with a 0.25° angular resolution. Its sensing range can go up to 80 m with an error of about 1 cm at 30 m. We have implemented a

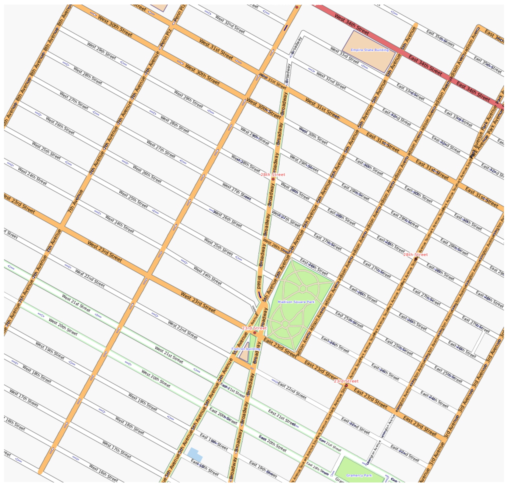

Fig. 6. Section of Manhattan between longitudes $-73.9975°$ and $-73.9812°$ and latitudes $40.7378°$ and $40.7496°$.

dynamic object detection and tracking method similar to the one proposed in [15] where *synthetic scans* are created and areas of difference between consecutive scans are tracked by particle filters. Figure 5 shows our tracker in action on two vehicles where the estimated vehicle bodies lie close to the raw points returned by the laser rangefinders.

B. *Experiment I*

For this first experiment, we propose to test the accuracy of our tracker in a heavy traffic environment. We load a small part of the Manhattan (New York) road network into our simulator (visible on Figure 6) and in which we initialize 1000 vehicles (modeled with the rail-based model presented in Section II-E2). In this experiment, we use only the laser rangefinder placed in front of the intelligent vehicle, pointing forwards. The intelligent vehicle wanders randomly in the city during the period of an hour and we record how many cars are visible within the sensor range and how many cars are detected and successfully tracked.

In this time period, our vehicle drove 25 km, encountered 663 other vehicles and successfully tracked 380 of them. We also computed the measured distance and azimut to the tracked vehicles and measured errors of 0.2114 ± 1.535 m and 0.1478 ± 0.1894 rad respectively.

C. *Experiment II*

Assuming we are satisfied with the performance of our vehicle tracker, we decide to use it to perform *platooning* on highways. *Platooning* is a complex task that requires automobiles to be able to drive in a controlled and coordinated fashion. In this second set of experiments, we will use four intelligent vehicles with four active SICK LMS 291 sensors. For each experimental run, our vehicles are placed in a highway ring. This three-lane ring consists of 4 segments:

- a 1000-meter-long straight segment followed by
- a curve with a radius of 500 meters followed by
- another 1000-meter-long straight segment and

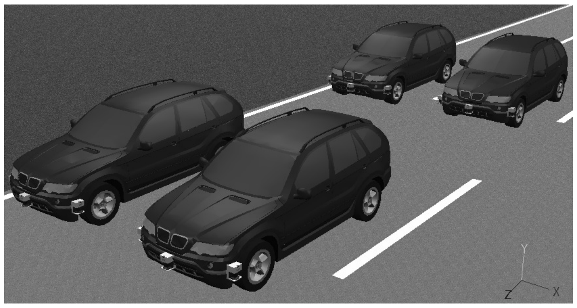

Fig. 7. Screenshot of our simulation showing four intelligent vehicles performing *platooning* on a highway.

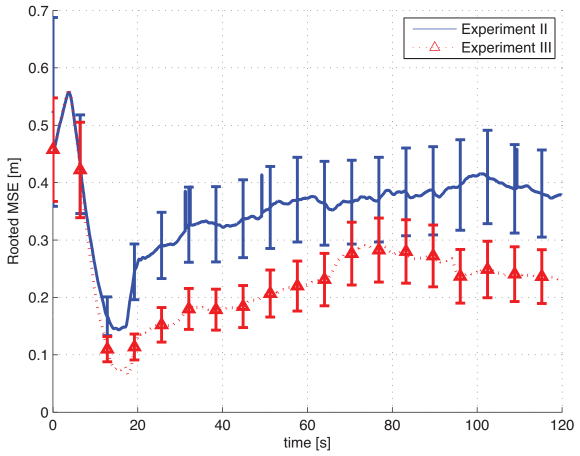

Fig. 8. Average and 95% confidence intervals of the rooted mean square error of the distances between each car and the desired distances depending on time for experiments II and III.

- a curve with a radius of 500 meters.

resulting in a 5142-meter-long ring. Our cars have to perform a rectangular formation (10×3.5 meters) at a speed reaching 120 km/h. Figure 7 shows an example of such a formation.

We assume that the cars are able to monitor the lane markings. Hence, they can use the PI controller mentioned in Section II-D to track the lane centers. To realize the formation, vehicles still need to gather the distance and bearing to the other vehicles and apply a tuned PID controller. The position of each car is monitored during a run and each run lasts 2 minutes. After 100 runs, the average rooted mean square error (MSE) between the actual distances between each pair of cars and the desired distances is computed.

Figure 8 shows the average rooted MSE over all runs. We can observe that the formation is stable and that the average error stays around 40 cm.

D. *Experiment III*

In a third experiment, we allow the cars of the previous experiment to communicate their current speed (through a realistic 802.11 Wi-Fi simulated using OMNeT++). This additional information is integrated into our tracker by giving a higher weight to particles whose speed is closer to reality. Therefore, we obtain a more accurate estimation of both the range and the bearing to other vehicles. The new errors on the range and bearing are now equal to 0.0164 ± 0.8077 m and 0.0163 ± 0.0986 rad respectively.

Figure 8 shows the average rooted MSE over all runs. The rooted MSE is sensibly lower than the one of the previous experiment and stabilizes between 25 and 30 cm.

E. Discussion

We showed in this section how the design and implementation of intelligent steering algorithms can be accelerated using our framework. The examples shown here represent a small fraction of the available potential. In a similar fashion, numerous other aspects of intelligent transportation systems can be tested in realistic traffic conditions and their effects analyzed and validated. A non-exhaustive list of these aspects includes driver assistance, change in the physical vehicle design, extension of sensing capabilities, increase in the drivers reaction time, change in the road network topology and introduction of cooperative behaviors.

IV. Conclusion

We presented a realistic traffic simulator plugin for Webots, a realistic robotics simulator. As this plugin is mainly based on the Open Dynamic Engine, it is also compatible with the open-source Player/Gazebo [24] platform. This plugin is well suited for the evaluation of intelligent vehicles on realistic road networks with realistic road partners. Multiple intelligent vehicles can be tested at the same time with different controllers and sensors. We illustrated some of the features of this simulator on three simple scenarios.

Future works include systematic validation against different vehicle models, as well as comparison with further simulation platforms and real data.

Additional Material

Source code and videos are available on http://disal.epfl.ch/research/context_aware_its/simulator/.

References

[1] A. Byrne, A. de Laski, K. Courage, C. Wallace, *Handbook of computer models for traffic operations analysis*, Technology Sharing Report FHWA-TS-82-213, 1982.

[2] H. Payne, *Models of freeway traffic and control. Mathematical Models of Public Systems*, Simulation Council Proceedings Series, Vol. 1, No. 1, pp 51-61, 1971.

[3] Sumo, *http://sumo.sourceforge.net/*

[4] TraNS, *http://trans.epfl.ch/*

[5] Mechanical Simulation, *Carsim Reference Manual Version 8*, Mechanical Simulation Corporation, Ann Arbor, 2009.

[6] R. Wiedemann, *Simulation des Verkehrsflusses*, Schriftenreihe des Instituts für Verkehrswesen, University of Karlsruhe, 1974.

[7] G. Cameron, G. Duncan, *PARAMICS-Parallel microscopic simulation of road traffic*, The Journal of Supercomputing, 1996, Vol. 10, No. 1, pp. 25-53.

[8] L. Elefteriadou, J. Leonard, H. Lieu, G. List, *Beyond the Highway Capacity Manual: A Framework for Selecting Simulation Models in Traffic Operational Analyses*, Transportation Research Board Annual Meeting, 1999.

[9] J. Barceló, J. Casas, *Dynamic network simulation with AIMSUN*, Int. Symposium on Transport Simulation, 2003.

[10] DARPA Urban Challenge: http://www.darpa.mil/GRANDCHALLENGE.

[11] SiVIC, *http://www.civitec.net/*.

[12] O. Michel, *Webots: Professional mobile robot simulation*, Journal of Advanced Robotic Systems, 2004, Vol.1, No.1, pp. 39-42.

[13] A. Varga, *Software Tools for Networking: OMNeT++*, IEEE Network Interactive, Vol. 16, No. 4, 2002.

[14] C. M. Cianci, J. Pugh, A. Martinoli, *Exploration of an Incremental Suite of Microscopic Models for Acoustic Event Monitoring Using a Robotic Sensor Network*, IEEE Int. Conf. on Robotics and Automation, pp. 3290-3295, 2008.

[15] M. Montemerlo et al., *Junior: The Stanford Entry in the Urban Challenge*, Journal of Field Robotics, pp. 569-597, 2008.

[16] A. Broggi, P. Cerri and P. Antonello, *Multi-Resolution Vehicle Detection using Artifical Vision*, IEEE Intelligent Vehicles Symposium, pp 310-314, 2004.

[17] OpenStreetMap, *http://www.openstreetmap.org/*.

[18] Open Dynamic Engine, *http://www.ode.org/*

[19] J. L. Harned, L. E. Johnston, G. Scharpf, *Measurement of tire brake force characteristics as related to wheel slip (antilock) control system design*, SAE Technical Paper No. 690214, 1969.

[20] E. Bakker, H. B. Pacejka, L. Lidner, *A new tire model with an application in vehicle dynamics studies*, SAE Technical Paper No. 890087, 1989.

[21] A. J. Kotwicki, *Dynamic models for torque converter equipped vehicles*, SAE Technical Paper No. 820393, 1982.

[22] D. Helbing, A. Hennecke, V. Shvetsov, M. Treiber, *Micro- and macro-simulation of freeway traffic*, Mathematical and Computer Modelling, 2002, Vol. 35, pp. 517-547.

[23] M. Linderoth, K. Soltesz, R. M. Murray, *Nonlinear Lateral Control Strategy for Nonholonomic Vehicles*, American Control Conference, pp. 3219-3224, 2008.

[24] N. Koenig, A. Howard, *Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator*, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2149-2154, 2004.

[25] Y. Zhang, E. K. Antonsson and A. Martinoli, *Evolutionary engineering design synthesis of on-board traffic monitoring sensors*, Research in Engineering Design, Vol. 19, No. 2-3, pp. 113-125, 2008.

[26] Y. Zhang, E. K. Antonsson and K. Grote, *A new threat assessment measure for collision avoidance systems*, IEEE Intelligent Transportation Systems, pp. 968-975, 2006.