A Framework for the Evaluation of the Impact of Adaptive Cruise Control Systems on CO_2 Emissions in Microscopic Traffic Simulations

W. Maier[1], P. Schott[1] and F. Mauz[1]*

[1] Department of Mobility and Driver Assistance, Berner & Mattner Systemtechnik GmbH, D-80807 Munich, Germany, werner.maier@berner-mattner.com

Introduction

Vehicle emissions are becoming a serious issue, since they tremendously impair life quality especially in metropolitan areas. Hence, city authorities make a big effort to reduce vehicle emissions. Methods related to urban traffic management include green waves, which maintain the traffic flow on roads with heavy traffic controlled by traffic lights (Toffolo et al., 2014). Other concepts from the field of demand and access management aim at restricting the traffic inside certain areas, such as for example the city centre or the area inside the innermost ring road in big cities.

An ecologically aware traffic policy by the city authorities is one remedy to reduce emissions in urban areas. Another aspect is the development of economic fuel-saving vehicular technology. Over the past years, car manufacturers have been presented more and more efficient engines. The development of hybrid motors or even fully electric vehicles has decreased the amount of emissions issued by a vehicle to a minimum.

Besides, the amount of emissions produced by a vehicle also heavily depends on the acceleration behaviour of the driver. Frequent and harsh accelerations and braking manoeuvers lead to higher vehicle emissions than rare and soft accelerations. Hence, drivers should strive for flat speed profiles. A system which can assist the driver in that is Adaptive Cruise Control (ACC). This system automatically controls the speed of a vehicle subject to a target speed determined by the driver and subject to a target distance to the vehicle on front (Jurgen, 2006). The desired target distance to the vehicle in front can be influenced by the driver by setting a time gap. This time gap is a constant which defines the ratio of the desired safe distance and the current speed of the car. If a smaller value is chosen for the time gap by the driver, the safe distance is also smaller at a given vehicle speed and the vehicle in general gets closer to the vehicle in front. ACC systems with soft velocity controllers perform soft accelerations and, hence, are expected to reduce vehicle emissions.

In order to investigate the effect of ACC vehicles on emissions, we developed a software framework which allows for the simulation of a varying number of ACC vehicles in microscopic traffic scenarios. A weakness of most microscopic traffic simulators is that the models which are used to determine the vehicle dynamics have limitations. Furthermore, most microscopic traffic simulators do not include vehicles with Advanced Driver Assistance Systems (ADAS) such as ACC. On the other hand, vehicle simulators with very detailed models for the engine and the powertrain are usually designed for very small traffic scenarios with only a few vehicles.

Hence, we present a driver simulator in our framework which overcomes these limitations and allows for the simulation of ACC vehicles with a sophisticated powertrain model. Furthermore, the driver simulator is designed and realized in a way that it is able to simultaneously compute speed profiles for a large series of vehicles in a microscopic traffic scenario.

System Structure of the Simulation Framework

Figure 1 shows the system structure of the simulation framework proposed in this paper. The simulation framework links its two main components, a traffic simulator and a driver simulator, over a TCP/IP connection. The driver simulator consists of three components, the sensor model, the driver and vehicle models and a test and execution environment.

The sensor model simulates the Electronic Control Units (ECUs) in an ADAS vehicle which process data from radar sensors or stereo cameras. These ECUs are in charge of detecting a vehicle in front and provide estimates of its distance and its velocity. The block *driver and vehicle models* provides, on the one hand, models which describe the car-following behaviour of human drivers and, on the other hand, models which implement distance and velocity control mechanisms for the simulation of ACC systems. Furthermore, a vehicle powertrain model is part of the driver simulator so that the acceleration behaviour of the vehicles in the simulation is more realistic. Using the fuel consumption maps of the powertrain model, the amount of CO_2 emissions can be computed. The test and execution environment is provided by the tool MESSINA, a test platform for model-based ECU function development (Berner & Mattner, 2014). It offers a signal pool which simulates the bus system in a real-world vehicle and a test case submodule. The test case submodule allows for the parameterization of the traffic scenario or the driver / vehicle models. To this end, a number of tests

can be defined and executed for different traffic scenarios, traffic levels and penetration rates of ACC vehicles, which, at the end of the day, facilitates a scientific analysis of the impact of ACC on CO_2 emissions.

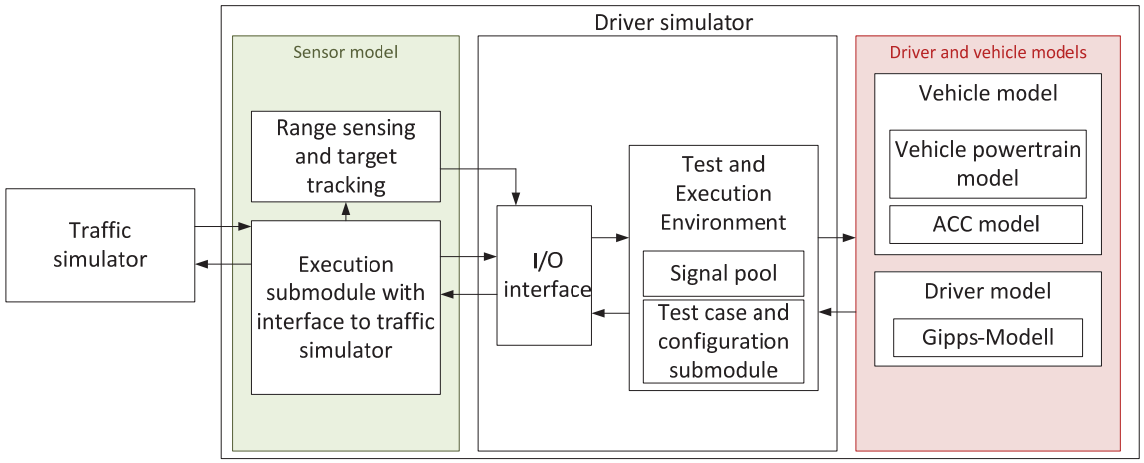

Figure 1: System structure of the simulation framework

The Components of the Driver Simulator

Figure 2 shows the interplay of the components of the driver simulator. The signal pool serves as a communication entity which enables the sensor model and the driver and vehicle models to exchange data during a simulation step. In the following, the components of the driver simulator are explained in detail.

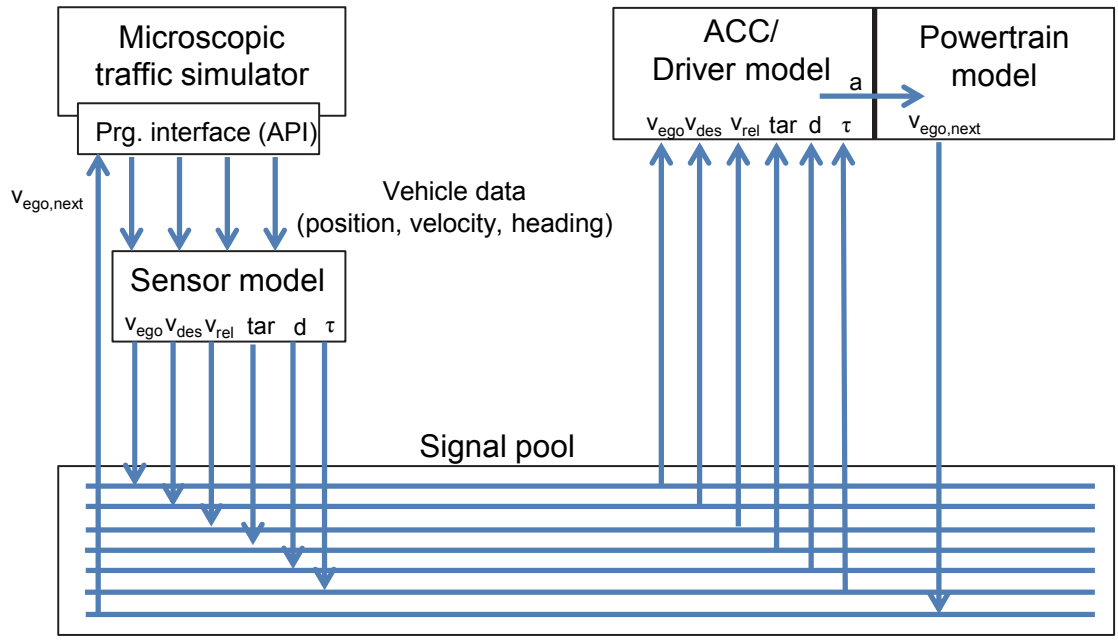

Figure 2: Communication of the components of the driver simulator

Sensor model

The sensor model has access to relevant data from the traffic simulation, such as the positions, the velocities and the headings of all vehicles, using a programming interface (API) provided by the traffic simulator. Since in our setup the simulation of the traffic scenarios is done by the simulation software SUMO (Simulation of Urban MObility (Krajzewicz et al., 2012), the corresponding API TraCi (Wegener et al., 2008) is used in our sensor model for the extraction of vehicle data.

The sensor model simulates the environment sensor of a given vehicle from the simulation (*ego vehicle*). This sensor can be a radar or a lidar sensor. Depending on the type of sensor, the angle of aperture and the range for the target vehicle detection varies. The sensor model is parameterized in a way that these sensor characteristics are taken into account. In the overall system, the sensor model has the task of detecting whether a target vehicle is available. This information is published on the signal pool in the Boolean variable *tar*. If this is the case, the values for the relative speed of the target vehicle w.r.t. to the ego vehicle v_{rel} and for the distance from the ego vehicle to the target vehicle *d* are calculated. The relative speed and distance to the target vehicle are the main control variables of the ACC model. Furthermore, the current vehicle speed v_{ego} and the speed desired by the driver v_{des} are relevant for the ACC / Driver model and thus published on the signal pool. The desired velocity v_{des} is usually derived from the maximum allowed speed on a road stretch defined by the speed limit. However, when modelling calm or aggressive human drivers, a negative or a positive small offset can be added to the speed limit, respectively. The parameter τ is the time gap for the ACC system set by a driver. As another relevant input to the ACC model for the safe distance, it is also published on the signal pool.

The range of the sensor is simplified and hence considered as a triangle. The three corners are calculated from the vehicle position, the heading of the vehicle, the sensor range and the angle of aperture of the sensor. They are stored in conjunction with each vehicle entity. After the coordinates are determined, it is verified whether there is a car within the triangle. Recognizing, whether a vehicle is on the same track and thus relevant, represents the major challenge, as it strongly depends on the geometry of the road and the road network. In the following, the two procedures *Target vehicle detection* and *Lane detection* performed by the sensor model are described.

Target vehicle detection: The target vehicle detection is done in four steps, which are visualized in Figure 3.

For every vehicle on the map, the following steps are performed:

1. Range check

The distances of the vehicles w.r.t. the ego vehicle are compared to the range of the sensor of the ego vehicle. Only vehicles within this range will be considered as potential target vehicles. To avoid checking all vehicles on the map, the map is divided into regions. The exact procedure is described below (cf. *Classification in grid regions*).

2. Aperture check

In this step it is verified whether the potential target vehicle is inside the sensor triangle and, hence, inside the viewing angle of the sensor of the ego vehicle.

3. Lane check

The next step is to check whether the vehicles inside the field of view of the sensor are relevant for the ego vehicle, which means that they are on the same lane. Figure 3 shows that the lane detection is relatively simple if the road is straight. However, a more sophisticated method, as described below (cf. *Lane detection*), might be necessary in case of complex road geometry. After the lane check, there are two vehicles in the scenario depicted in Figure 3, which come into consideration as being the target vehicle.

4. Proximity check

The final selection of the target vehicle is made by examining, which of the considered vehicles still has the smallest distance to the ego vehicle. In the case illustrated in Figure 3, the vehicle highlighted by the red circle is identified as the target vehicle by the described procedure.

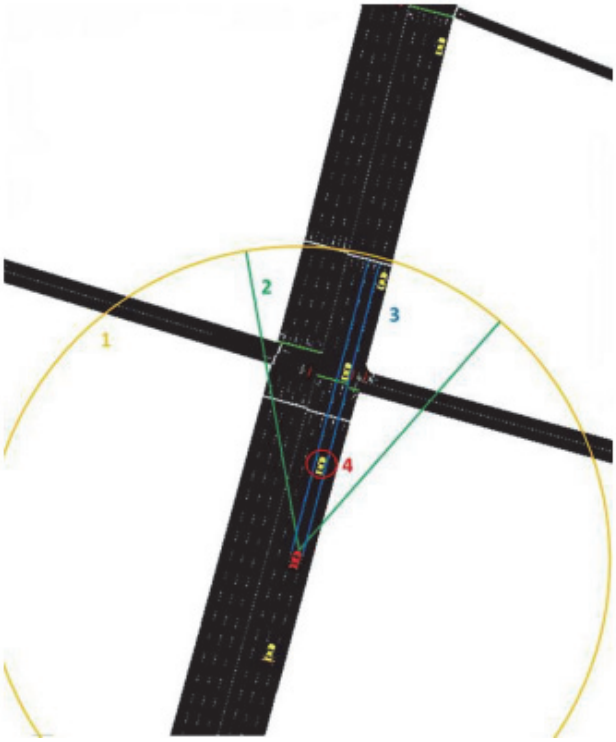

Figure 3: Target vehicle detection

Lane detection: In the procedure for lane detection (cf. step 3 of *Target vehicle detection*), different cases have to be distinguished. Considering the traffic simulator SUMO, parameters from the map of the traffic scenario can be read to detect the relevant track. These are the road link ID, the track ID, the track width, the link shape, the number of follower road links and the names of the follower links.

In a first step, all vehicles of the oncoming traffic or entering at crossroads can be excluded as target vehicles using the direction in which a vehicle is moving. The following cases are distinguished in the lane detection procedure used in this work:

1. Ego vehicle and possible target vehicle are on the same lane.

If both vehicles are on the same lane, the possible target vehicle is on the relevant lane and thus the target vehicle. This may be checked using the trackID. The trackID can be queried for each vehicle using the API and is the firm identifier of the track.

2. Ego vehicle and possible target vehicle on line.

The direction of travel of the vehicle can be identified by its heading. Due to the direction it can be detected, whether or not two cars are moving in the same direction on a straight road. If this is the case, the preceding vehicle is the target vehicle, if it is located in a rectangle with the dimensions of the track width and range of the sensor in front of the ego vehicle. The track width can be queried for each trackID. In Figure 3 the corresponding rectangle is drawn in blue.

3. Ego vehicle and possible target vehicle in a curve.

The most challenging task is, as in reality, to find and track the correct target vehicle in curves. This task is even more complicated if there is the possibility to leave the road using a side street. For this approach the number of follower road links, the names of the follower links and the shape of the link are used.

a. Without the possibility to leave the road

The simpler case is that there is no possibility to leave the road within the range of the sensor. In this case it needs to be checked whether the potential target vehicle is on the relevant road link, which is done by querying the next link which is attached to the end node of the link where the ego vehicle is currently going. It needs to be checked if there is only one link attached. Afterwards, using the road link ID of the potential target vehicle it is checked whether the vehicle is on this link. If this is not the case, the link which is attached to the next link is considered. This procedure is continued until a road link is no longer within the range of the sensor.

b. With the possibility to leave the road

This is the most challenging case because the ego vehicle and the possible target vehicles have the opportunity to leave at crossroads or branches. The principle applied in the previous case is not practicable anymore, once there are several links following one link. Now it must be decided which of the following links is the relevant one for the ego vehicle. Therefore, the link is selected, which forms part of the most probable route the vehicle will follow. Considering geometric characteristics, the most probable route usually goes along straight lines. Hence, the link with the least deviation in terms of direction w.r.t. the current link is chosen. The alignment of the links can be determined by their shapes.

Regarding the used methodology, it is possible, particularly in urban areas, that a target vehicle is lost for a short time, for example at curves. Comparable difficulties occur in reality, too, as vehicle detection is a complex task. Due to the used methodology, in almost all cases, the current vehicle is found and identified again quickly, if it temporarily has been lost.

Classification in grid regions: The speed of a simulation strongly depends on the duration of one simulation step. The procedures of the sensor model, as they are described in this section, are computationally quite expensive. This is because the sensor values of the model need to be determined for each vehicle on the map. In order to determine the respective relevant vehicle, the distance to any other vehicle must be calculated theoretically for each vehicle on the map to recognize whether it is within the range of the sensor and the driver's visibility range. Thus, the calculation time increases in a quadratic way with the number of vehicles. To reduce the calculation time of the sensor model, the map is divided in a grid (see Figure 4) and a two-dimensional array of the size of the grid.

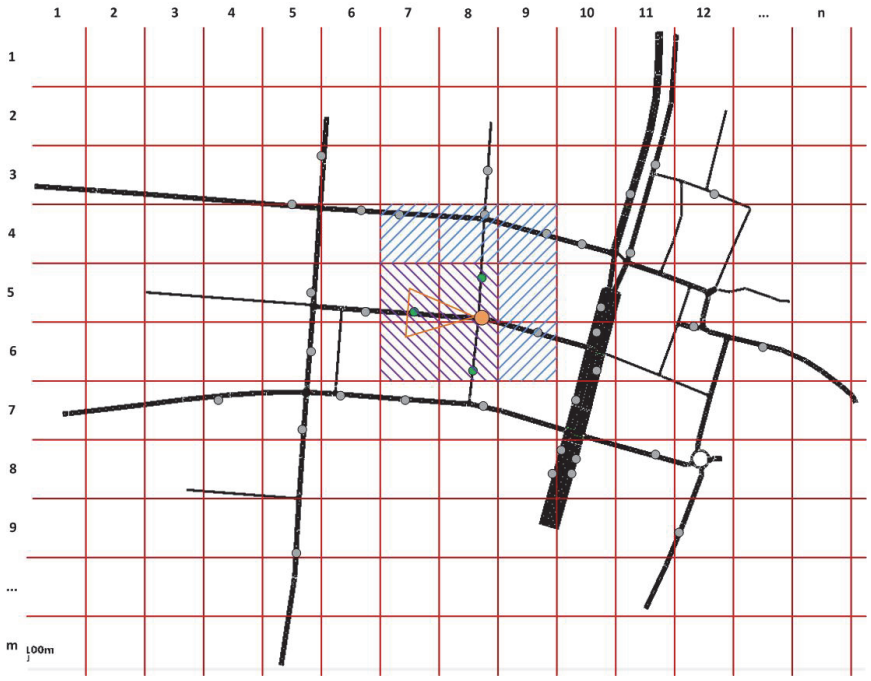

Figure 4: Classification of grid regions in a city traffic scenario

The grid consists of rectangles whose lengths correspond precisely to the range of the sensor. Based on the position of the vehicle, it is determined in which field it is located. In addition, at the beginning of each call, the sensor model iterates over the vehicles and each vehicle is stored in the corresponding field. For the distance calculation, only the field in which the vehicle is located and the eight adjacent squares are of interest. In a subsequent step, based on the direction of the vehicle, it is possible to limit the relevant fields further. Thus, only two to four fields are relevant, depending on the driving direction and the angle of aperture of the sensor. The vehicles in the relevant fields are read from the corresponding fields of the array and put together in a list. In the calculations of the sensor model, only the vehicles in the created list are considered. By doing this, the simulation time can be reduced significantly.

Figure 4 shows a schematic representation of the grid layout of a city scenario. The circles represent the vehicles. The largest circle is the considered ego vehicle. The hatched rectangles around the ego vehicle are the relevant rectangles which were filtered out in the first step. The radar triangle of the ego vehicle is also shown. Due to that, the four relevant rectangles are found. They are indicated by an altered hatching in the figure.

ACC / Driver model

The ACC / Driver model controls the velocity of the vehicle by producing an acceleration value subject to the input values for v_{ego}, v_{rel}, v_{des}, *tar*, *d* and τ. The block diagram of the ACC / Driver model is designed in MATLAB/Simulink. For the integration of the model in the test and execution environment MESSINA, C-code is automatically generated from the MATLAB/Simulink model and compiled into a Dynamic Link Library (DLL). The ACC / Driver model offers the possibility to simulate both the car-following behavior of an ACC system and the car-following behavior of a human driver. The test and execution environment indicates for each vehicle which model to use (ACC or Driver).

In case of the simulation of an ACC vehicle, the ACC / Driver model contains two modules, one for Cruise Control and one for Adaptive Cruise Control. The Cruise Control unit is active if the variable *tar* indicates that there is no vehicle in front within the range of the sensor system of the vehicle. Then the vehicle can accelerate under free-flow conditions and control the velocity in a way that the vehicle goes at the velocity desired by the driver v_{des}. The structure of the Cruise Control unit is quite simple and shown in Figure 5. A PI-controller calculates the new velocity from the difference between the desired velocity v_{des} and the velocity of the ego-vehicle v_{ego}. The acceleration a_{CC} which results from the new velocity is calculated by derivation. The parameters of the unit are the proportional and integral values of the PI-controller.

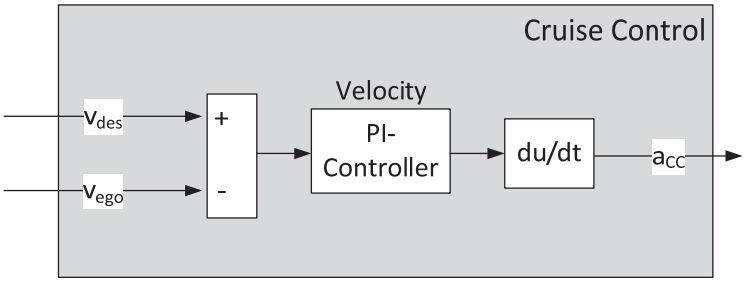

Figure 5: Structure of the Cruise Control unit

In the car-following case the Adaptive Cruise Control unit is active. Its structure is shown in Figure 6. It consists of three PI-controllers: one for adjusting the distance to the target vehicle and two for adjusting the velocity. The PI-controller for the distance calculates a velocity to adjust the desired distance to the target. The desired distance to the target is calculated by the current velocity v_{ego} and the desired time gap τ. The realization of the calculated velocity is the task of the two velocity controllers. The two controllers for velocity are parameterized in a different way. The controllers are activated subject to the calculated value of the PI-controller for distance. In cases where the actual distance of the vehicle to the target vehicle is close to the desired distance, the first velocity controller with a high damping is used. In contrast, the second velocity controller, which has a high control loop gain, is used if the actual distance to the target vehicle strongly deviates from the desired distance.

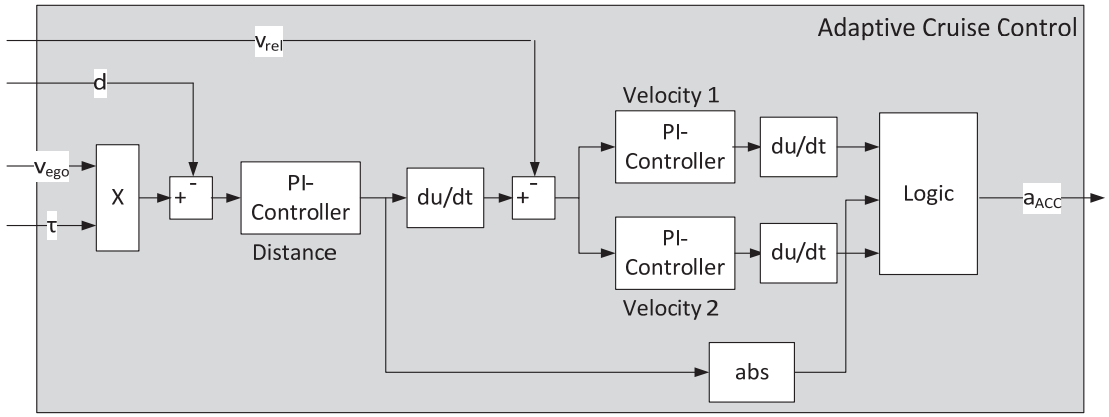

Figure 6: Structure of the Adaptive Cruise Control unit

As a car-following model for a human driver, the Gipps model (Gipps, 1981) is implemented in our system. Figure 7 shows the block diagram in MATLAB/Simulink. Similarly to the two-part automatic velocity control mechanism (Cruise Control or Adaptive Cruise Control), the Gipps model also distinguishes two cases – the free flow case (Free Flow Control Path) and the car-following case (Follow Control Path). The parameters A_E, B_E, B_T and θ characterize the driver type (aggressive, normal, calm) and can be set accordingly. To avoid collisions, the smaller velocity of the two paths (v_1 or v_2) is chosen as the output velocity. The velocity is integrated over time, which produces an acceleration value. Subject to the selected model (ADAS or Driver) and subject to the current situation (free-flow or car-following), the ADAS / Driver model provides an acceleration value as an output which comes either from the Cruise Control unit, from the Adaptive Cruise Control unit or from the Gipps unit.

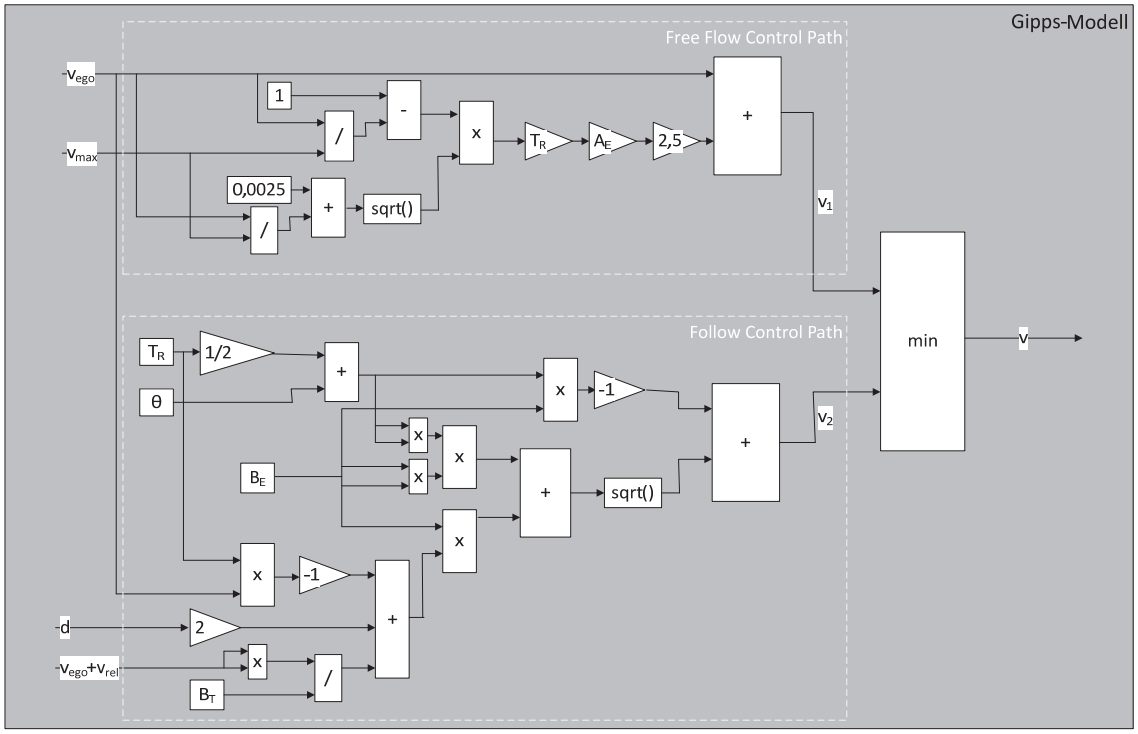

Figure 7: Block diagram of the Gipps model implemented in our ADAS / Driver model

Powertrain model

To make traffic simulations more realistic, the vehicles have to show a physically correct driving behavior, which is determined by a model for their powertrains. A powertrain model also facilitates accurate computation of instantaneous vehicle emissions $e_{ml/s}$.

As an input the powertrain model receives the target acceleration *a* which the vehicle is supposed to perform according to the computation inside the ACC or Gipps model. The powertrain model provides at the output a velocity value $v_{ego,next}$ which the vehicle is eventually able to perform, taking into account the physical and mechanical constraints inside its engine. Hence, the powertrain model makes a correction of the target acceleration in case unrealistic accelerations should be performed by the vehicle according to the ADAS / driver model.

Figure ***8*** shows the basic scheme of the powertrain model which is developed for the evaluation framework and applied in the simulations. Based on the driving resistance the engine power which is necessary to achieve the desired target acceleration is determined. From the acceleration the vehicle behavior and the emissions can be derived.

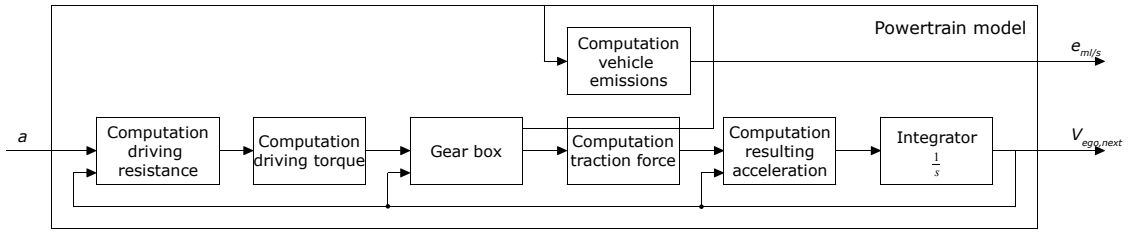

Figure 8: Schematic overview of the powertrain model

The driving resistance encompasses the rolling resistance, the resistance due to slopes, inertia and air drag. These types of resistance are computed individually and summed up. The vehicle engine has to generate a force which is as large as the force which results from the sum of these resistances to accelerate the car.

From this force the driving torque which the engine has to produce to achieve the desired acceleration is computed. Due to the mechanical characteristics of the engine it might happen that the engine is not able to produce a torque as large as required by the ACC controller or the Gipps model. This limitation in real-world vehicle engines is reflected in this calculation step of the powertrain model. The gear box selects the gear which is most efficient in terms of emissions based on the engine power, the engine speed and a fuel consumption map for the vehicle.

Afterwards, the traction force is determined from the engine torque, the gear ratio and the wheel dimensions. The acceleration of the vehicle results from the traction force. Due to constraints on the engine torque the resulting acceleration might be smaller than desired by the ACC controller or the Gipps model. A final temporal integration step then provides the velocity $v_{ego,next}$ which the vehicle performs in the next simulation step. This velocity value is fed back to the micro-traffic simulator.

The micro-traffic simulator and the driver simulator in **Figure 1** exchange data in each simulation step via the sensor model and the powertrain model. Hence, there is a continuous and steady processing loop, which allows for the simulation of ACC vehicles in a traffic simulation on-line. Our test and execution environment MESSINA permits the transmission of byte arrays over the wires of the signal pool. It is possible to transmit up to 960 bytes over one wire. If 4 bytes are allocated for a given data type (v_{ego}, v_{des}, v_{rel}, *tar*, *d*, τ or $v_{ego,next}$), the data of 240 vehicles can be transmitted over one wire in one simulation step. A larger number of vehicles in the simulation is supported by our simulation framework if multiple wires are allocated for the transmission of a given data type. If 20 wires are allocated per data type, e.g., 4800 vehicles can be handled in the simulation, which is usually sufficient for microscopic traffic scenarios.

Simulation Results

For the evaluation of the impact of ACC systems on CO_2 emissions in microscopic traffic scenarios, the penetration rate of ACC vehicles is increased in discrete steps (20%, 40%, 60%, 80% and 100%) and the amount of emissions produced by the vehicles at these penetration rates is compared to the amount of emissions produced by the vehicles in a scenario without any ACC systems at all.

To this end, a traffic scenario is considered, which covers a part of an urban ring road. The maximum allowed speed on the ring road is mainly at 60 km/h. Especially in the rush-hour traffic the urban ring road is heavily loaded. The considered scenario consists of the actual ring road and some entrances and exits. Looking at Figure 9, in addition to the road courses, the starting and ending points of the traffic flows are indicated.

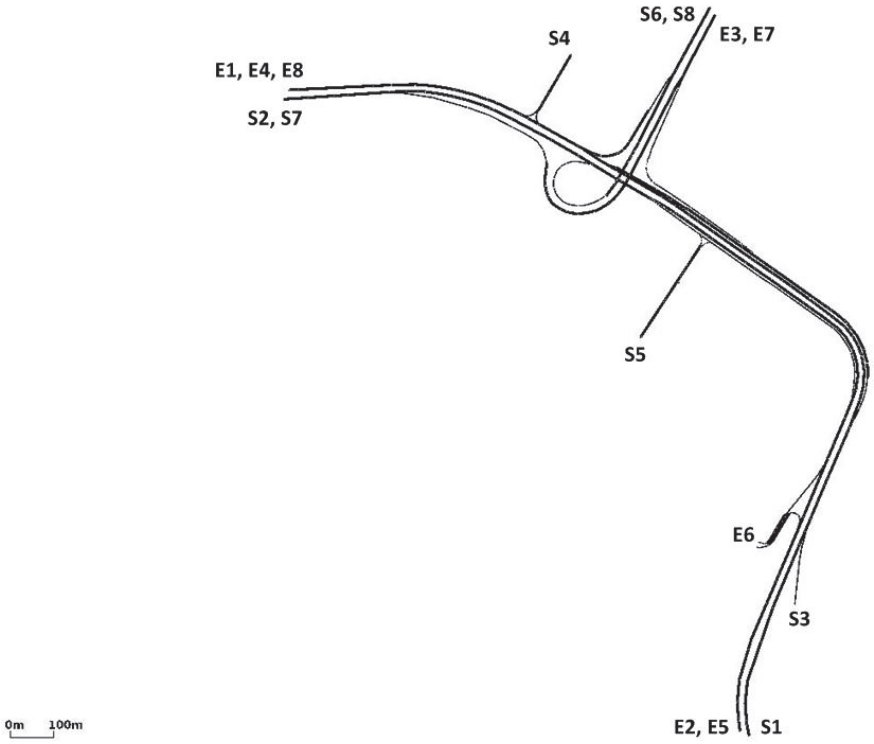

Figure 9: Traffic scenario of the urban ring road with the routes of the vehicles

The traffic flows are defined by their routes and numbers of cars. For the urban ring road, eight traffic flows are defined. The parameters for each traffic flow are listed in Table 1. In the column *connection*, the starting and end points of the different routes are listed, which also can be found in Figure 9. HV1 and HV2 represent the bulk of the traffic which runs through the urban ring road. The rest, which is flowing into the bulk of the traffic, is implemented in NV1 to NV6. On each individual traffic flow, there is a fixed proportion of all vehicles that is related to the traffic density on the route. This percentage can be read in the last column of Table 1.

Table 1: Traffic flows of the urban ring road scenario

Traffic flow	Route	Connection	Length of route in m	Proportion of vehicles in %
1	HV1	S1-E1	2418	34
2	HV2	S2-E2	2408	33
3	NV1	S3-E3	1654	2
4	NV2	S4-E4	727	2
5	NV3	S5-E5	1530	5
6	NV4	S6-E6	1994	2
7	NV5	S7-E7	1401	11
8	NV6	S8-E8	1166	11

Two variants are simulated for the urban ring road scenario, which differ in the traffic density. Variant 1 shows a lower traffic density than variant 2. In variant 1 there are 100 vehicles in the traffic scenario which enter the scenario at the beginning of the simulation with a frequency of one vehicle per 10 seconds. In contrast, in variant 2 there are 200 vehicles in the traffic scenario which enter the scenario at the beginning of the simulation with a frequency of one vehicle per 5 seconds.

Table 2 shows the results obtained for variant 1, for each considered penetration rate (level of equipment ACC). The average fuel consumption of individual vehicles can be reduced by the use of ACC at the maximum penetration rate by 0.5l/100km, which corresponds to a relative reduction of CO_2 emissions of 8.86 % w.r.t. the basecase scenario with 0% ACC vehicles.

Table 2: Results ACC, urban ring road, traffic density variant 1

Level of equipment ACC [%]	Total fuel consumption [ml]	Average fuel consumption [l/100km]	CO_2 emissions [g]	Relative reduction of CO_2 [%]
0	22554	5.47	59769	-
20	22211	5.38	58860	1.25
40	21778	5.28	57713	3.44
60	21350	5.17	56578	5.34
80	21081	5.11	55866	6.53
100	20557	4.98	54476	8.86

The velocity profiles of vehicles which are not equipped with an ACC system in the simulation are computed by the Gipps model in the ADAS / Driver model. It is assumed that vehicles without ACC are driven either by a calm, an average or an aggressive driver. In our simulations we choose equal shares of these driver types for all non-ACC vehicles. The parameters of the Gipps model in Figure 7 are set as given in Table 3 for these driver types. v_{max} is the speed limit of a given road stretch.

Table 3: Gipps parameter values for different driver types

Parameter	Calm driver	Average driver	Aggressive driver	Unit
A_E	3	5	8	m/s^2
B_E	2	3	5	m/s^2
B_T	3	5	8	m/s^2
θ	2	1.5	1	s
v_{des}	$v_{max} - 1.5$	v_{max}	$v_{max} + 1.5$	m/s

Defining certain penetration rates of ACC vehicles and certain shares of driver types in the test cases of our test and execution environment MESSINA means that the ACC vehicles and driver types are randomly distributed across the vehicles in the traffic scenario. It might happen that due to a rare random constellation all ACC vehicles go one after another in one line, which might have particular effects on the CO_2 emissions measured in the scenario. Hence, to even out statistical effects, performing multiple runs and averaging is necessary. In our analysis 25 simulation runs are performed per penetration level. The results obtained for the fuel consumption and CO_2 emissions are averaged over all runs. Table 2 (and Table 4 below) show the arithmetic means for fuel consumption and CO_2 emissions. For the investigation at a level of equipment equal to 100% only one run is performed, because the results are deterministic. By the number of runs, statements with sufficient accuracy can be made for the present work.

For each level of equipment, the range of its result values is determined. Therefore, confidence intervals are used. By using the confidence intervals, the probability with which a value lies within a specific interval, can be specified. For the results of the present study, confidence intervals, indicating a 95 % coverage probability, are determined. That means, the actual value of a level of equipment is, with a probability of 95%, within the specified interval. The confidence intervals of the present study have a maximum deviation of 0.69% from the arithmetic mean.

In Figure 10 the achieved savings of CO_2 emissions from Table 2 (last column) are shown as a function of the penetration rate of ACC vehicles (level of equipment). As expected, the reduction increases with increasing the number of ACC vehicles. The reduction of CO_2 emissions runs approximately linearly with increasing the level of equipment.

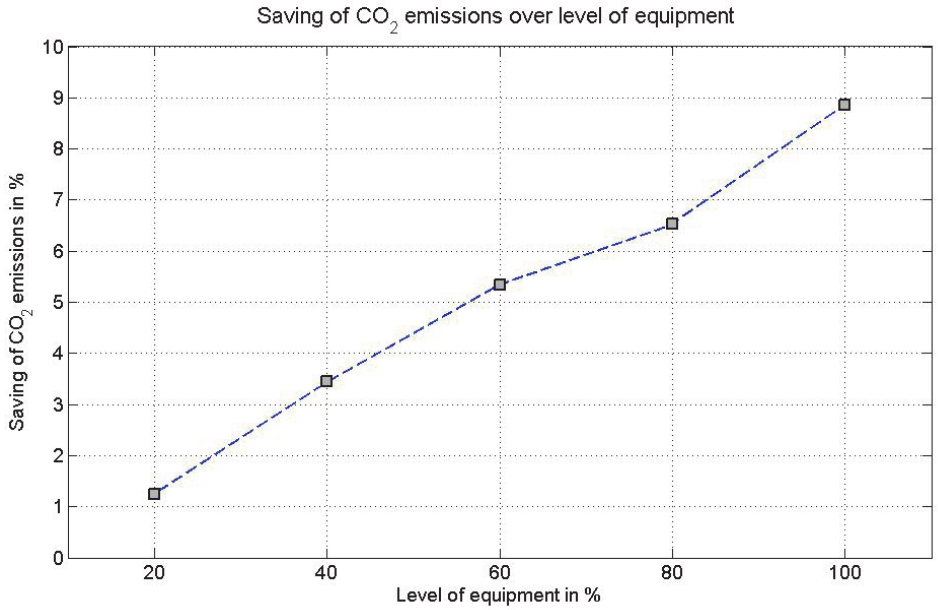

Figure 10: Results ACC, urban ring road, traffic density variant 1

In variant 2, the ACC system is examined at a higher traffic density. The results are shown in Table 4. It should be noted that, by a higher entry rate of vehicles on the map, the traffic situation will change. The additional traffic streams, which are flowing into the main traffic stream, have a higher traffic density, just like the main traffic stream.

Table 4: Results ACC, urban ring road, traffic density variant 2

Level of equipment ACC [%]	Total fuel consumption [ml]	Average fuel consumption [l/100km]	CO_2 emissions [g]	Relative reduction of CO_2 [%]
0	22628	5.48	59964	-
20	22280	5.40	59042	1.54
40	21978	5.33	58241	2.87
60	21618	5.24	57287	4.46
80	21206	5.14	56195	6.29
100	20950	5.08	55516	7.42

The savings generated by the ACC system in variant 2 are lower than in variant 1, but they still reach a saving of 7.42% at the maximum level of equipment. The savings from Table 4 (last column) are shown as a function of the penetration rate (level of equipment) in Figure 11.

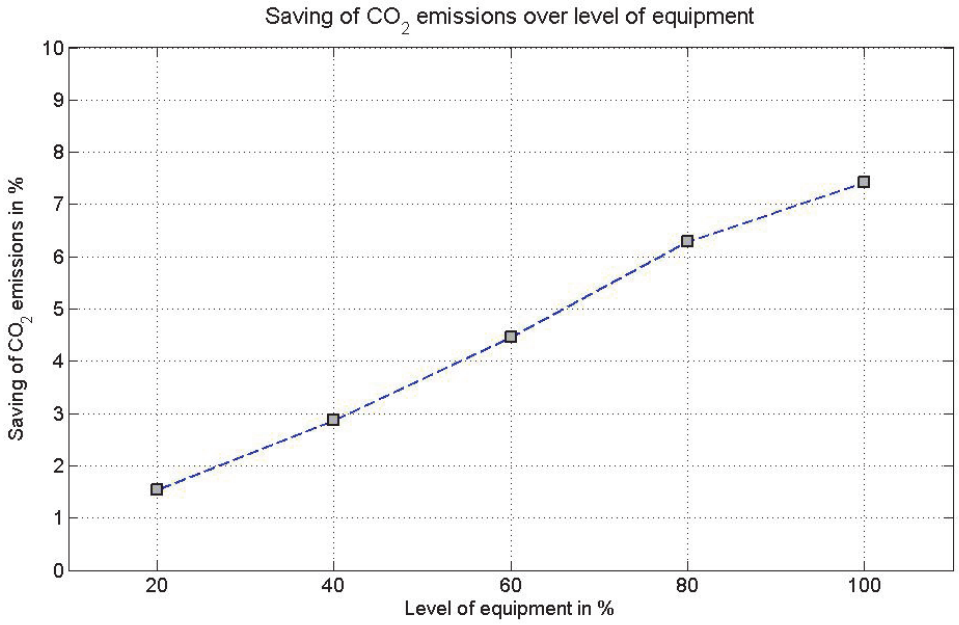

Figure 11: Results ACC, urban ring road, traffic density variant 2

Conclusion

In this work we present a framework for the evaluation of the impact of ACC vehicles on CO_2 emissions in microscopic traffic scenarios. The core of this framework is the test and execution environment MESSINA. Its signal pool serves as a communication entity for a sensor model, an ACC / Driver model as well as a powertrain model, which are all integrated in MESSINA via adapters. The framework establishes an on-line processing loop between the traffic simulator SUMO and a proprietary driver simulator developed for this framework. In this loop vehicle data is extracted from the microscopic traffic scenario, processed by the driver and vehicle models and fed back to the microscopic traffic simulation at each simulation step. The proposed framework provides realistic ADAS and powertrain models and is able to apply these models to a large number of vehicles in a traffic scenario.

The simulation results obtained in this work show that increasing the penetration rate of ACC vehicles in an urban ring road scenario leads to a reduction of CO_2 emissions in the traffic scenario. In case of a high level of equipment more platoons of ACC vehicles are formed, which has a calming effect on the flow of the traffic. Due to the damping effect of ACC the high-frequent longitudinal dynamics of individual vehicles in a platoon decrease. Acceleration and deceleration peaks are attenuated in the ACC platoon. Hence, a lower fuel consumption level results for the vehicles in the rear of a platoon.

In a ring road scenario with a higher traffic density ACC systems also exhibit a reduction of CO_2 emissions compared to a basecase scenario with no ACC vehicles. However, the effect on the fuel consumption is not as strong as in case of low traffic densities, since the desired constant vehicle movement is disturbed by the influence of the added road users. In case of a low traffic density the vehicles are able to cover a longer distance without the influence of other vehicles. Therefore, lower average consumption and higher savings result for the ACC system in scenarios a with lower traffic density.

Outlook

Future work in the development of our simulation framework will focus on the development of links to other microscopic traffic simulators, such as Aimsun, which provide different possibilities to overwrite the internal car-following model.

Moreover, the powertrain model is supposed to be extended such that it can handle multiple vehicle types in a traffic scenario. Currently, the parameterization of the powertrain model is static and simulates a middle-class passenger car. In the future, we plan to integrate parameter sets of several vehicle classes (small passenger cars, limousines, vans etc.). We also plan to extend the test engine of MESSINA in a way that the different vehicle types in the powertrain model can be distributed over the vehicles in the microscopic traffic scenario. Hence, different fleet compositions can be simulated in the traffic scenarios with defined shares of the vehicle types.

Furthermore, a link to external emission models will be developed.

Acknowledgement

This work is supported by the European Commission within the FP7 project ICT-Emissions, see also http://www.ictemissions.eu.

References

Berner & Mattner (2014), MESSINA: Test Platform for Model-based ECU Function Development, http://www.berner-mattner.com/en/berner-mattner-home/products/messina/index.html.

Gipps P.G. (1981), A Behavioural Car-Following Model for Computer Simulation, *Transport Research Board B*, 15, 105-111.

Jurgen R.K. (Ed.) (2006), *Adaptive Cruise Control*, SAE International.

Krajzewicz D., J. Erdmann, M. Behrisch and L. Bieker (2012), Recent Development and Applications of SUMO – Simulation of Urban MObility, *International Journal On Advances in Systems and Measurements*, 5, 128-138.

Toffolo S., E. Morello, Z. Samaras, L. Ntziachristos, C. Vock, W. Maier and A. Garcia-Castro (2014), ICT-Emissions Methodology for Assessing ITS and ICT Solutions, *Proc. Transport Research Arena*.

Wegener A., M. Piorkowski, M. Raya, H. Hellbrück, S. Fischer and J.-P. Hubaux (2008), TraCI: A Framework for Coupling Road Traffic and Network Simulators, *Proc. 11th Communications and Networking Simulation Symposium*, 155-163.