Data-driven Approach for Accurate Estimation and Validation of Ego-Vehicle Speed

Adina Aniculaesei*, Meng Zhang* and Andreas Rausch*
*Institute of Software and Systems Engineering
TU Clausthal, 38678 Clausthal-Zellerfeld, Germany
Email: {adina.aniculaesei, meng.zhang, andreas.rausch}@tu-clausthal.de

***Abstract*—This paper proposes a data-oriented approach for the accurate estimation of the ego-vehicle speed. The approach combines long-term estimation with short-term estimation mechanisms to produce an accurate estimation of vehicle's tire circumference. The long-term estimation method approximates a standard value for the tire circumference on the basis of the vehicle configuration. In turn, the short-term estimation computes an estimation error for the tire circumference based on Global Positioning System (GPS) sensor data. The ego-vehicle speed is then computed on the basis of the estimated tire circumference and the current wheel speed measurement. In this approach, several error sources are considered: the GPS data, the road gradient and the rounding off of the estimated vehicle speed. The approach is validated on two real-world test data batches against the European New Car Assessment Programme (Euro NCAP) safety requirements. The results of the experimental validation demonstrate that the proposed vehicle speed estimation algorithm performs within the limits of the Euro NCAP requirements.**

Keywords–data-based multiresolutional learning; precise parameter estimation; automotive; ego-vehicle speed estimation; Euro NCAP requirements.

I. INTRODUCTION

Reactive systems and requirements defined upon them are getting increasingly complex. These systems, used to build a variety of applications, such as multimedia devices or avionic systems, exhibit stochastic behaviour and also operate under real-time constraints and constraints on other resources [1]. Ensuring the correct functioning of these systems is of paramount importance, especially for those systems deployed in safety-critical applications.

Through their continuous interaction with their operation environment, reactive systems are subject to a variety of external stimuli. Often reactive systems are required to perform parameter estimation based on the large amount of data received from the environment. Ideally, the input data is structured, independent and identically distributed. Furthermore, the system can access the data at any time and without any concerns for the required processing time or the storage space. Losing et al. [2] observe that real-world applications produce data in a streaming fashion at an increasing rate, requiring processing on a large-scale and in real-time, as well as continuous learning.

Reactive systems work often not only with input data perceived directly by the sensors from the system environment. Instead, such systems keep an internal state and preserve values of the state variables over several iterations. Architectures, which enable reactive systems the storage of data on a long-term basis but also offer the ability to react to current input coming from the environment, present a particular advantage. Such an architecture would resemble the human memory model, as indicated by Atkinson and Shiffrin [3] and Losing et al. [2].

In the automotive domain, every aspect of driving is supported to a larger or lesser degree by complex software systems. Such systems enhance the driving experience and increase the vehicle safety. A basic functionality introduced in automobiles at the begining of the 20^{th} century is the speedometer, which gains even more attention, especially in the context of Advanced Driving Assisstance Systems (ADAS) and autonomous driving. Car speedometers are reactive systems, which are confronted with the data-related problems mentioned previously.

The job of the speedometer is to indicate the instanteneous speed of the car in miles per hour, kilometers per hour, or both. The speed displayed on the speedometer is however not the actual speed of the vehicle, but an estimation of it. Thus, vehicle speedometers are not 100% accurate.

Car manufactureres build speedometers so that the estimated vehicle speed falls within a narrow range [4]. This range is usually specified through compulsory regulations. Consider for example the european laws [5], which impose the requirement in (1):

$$0 \leq v_{display} - v_{real} \leq 0.1 \cdot v_{real} + 4 \ \frac{km}{h} \tag{1}$$

under the precondition that

$$40 \leq v_{real} \leq 120 \ \frac{km}{h} \tag{2}$$

where $v_{display}$ is the speed displayed on the dashboard of the ego-vehicle, and v_{real} is the actual vehicle speed. Before its release, the vehicle speedometer is subject to an initial calibration, which depends strongly on the vehicle model and configuration. As long as the car is maintained according to its factory specifications, the speedometer should continue to work within its predefined range. Once a parameter in the system configuration is changed, the speedometer must be recalibrated. Consider for example when tires with a profile different from that mentioned in the factory specification are mounted on the vehicle [4]. The problem of the ego-vehicle speed estimation is an instance of a larger one, namely the problem of precise parameter estimation.

There are various approaches developed for the ego-vehicle precise parameter estimation in the automotive domain. These approaches use a combination of sensors to estimate as accurate as possible the motion of the ego-vehicle: laser range finder combined with monocular camera to estimate the ego-vehicle orientation and the scale, i.e. the length of the translation direction vector [6] and a combination of monocular and stereo cameras to estimate rotational and respectively translational movements of the ego-vehicle [7]. Other approaches use deep learning methods to process optical flow and depth estimation with a monocular camera in order to approximate the ego-vehicle speed [10].

 ISBN: 978-1-61208-771-9

These approaches present, however, various disadvantages. Laser range finders and cameras can be affected by unfavorable weather conditions. Algorithms relying on feature matching between consecutive frames usually use various mechanism to reject poor features from the second of two consecutive frames. One such mechanism is to apply optical flow backwards and reject those feature for which the distance between initial and computed position in the first frame is below a certain threshold [7]. However, such algorithms suffer, if poor weather conditions or significant illumination changes cause several consecutive camera frames to be unusable. Furthermore, optical flow can be problematic, if significant changes in illumination appear, e.g., too dark or too bright.

For the estimation of ego-vehicle speed displayed on the dashboard instrument, we propose a data-oriented approach. The approach builds upon the theoretical frameworks proposed in [3] and [2] and combines long-term and short-term estimation mechanisms for the accurate approximation of vehicle tire circumference. On one hand, the long-term estimation mechanism makes use of the characteristics defined in the vehicle configuration, which remain relatively constant over time. On the other hand, the short-term estimation mechanism uses GPS sensor measurements in order to derive a corrective value, which is then used to adjust the result of the long-term estimation procedure. We apply several filters in order to filter out poor or unreliable GPS data, which appears due to loss of signal in blocked areas or due to sudden acceleration or deceleration of the ego-vehicle.

The rest of the paper is structured as follows. Section III illustrates the overall approach of this paper, while in Section IV the realization of the presented concept is demonstrated on the case study of a speedometer model implemented in MATLAB/SIMULINK. In Section V, experimental validation of the proposed vehicle speed estimation approach is performed on real-world scenarios and results are discussed. Section VI concludes this paper and point out interesting future research directions.

II. Related Work

There are several works which focus on the problem of ego-vehicle parameters estimation. Huang and Stachniss [6] present an approach for ego-motion using a monocular camera together with a laser range finder. The approach uses the camera images to estimate the five degrees of freedom relative orientation and a variant of the iterative closest point algorithm with one degree of freedom to estimate the scale. Nedevschi et al. [7] use video data to increase the accuracy of the ego-vehicle motion estimation. The video data is processed through procedures for feature detection and filtering, optical flow and epipolar geometry in order to obtain the essential matrix, from which the rotation and the translation of the ego-vehicle are computed.

Lee et al. [8] use a multi-camera system with minimal field-of-views for ego-motion estimation. The camera system is modelled as a generalized camera and the motion of the ego-vehicle is constrained to the Ackerman motion model. The method is compared to the ground truth provided by GPS and Inertial Navigation System (INS) sensors. Qimin et al. [9] developed a method for computing vehicle speed on the basis of sparse optical flow obtained from image sequences. The proposed method identifies distinct corners in camera images and maps the feature set of one frame on the consecutive frame. The vehicle speed is computed as the average of all speeds estimated by every matched corner. The time of execution for one iteration is $59\ ms$, while the mean error of speed estimation relative to the GPS measurement is $0.121\ \frac{m}{s}$.

Rill [10] uses the intuition that the magnitude of optical flow is positively correlated with the speed of the moving observer to develop a method for ego-speed estimation. The presented approach applies deep neural network based optical flow estimation and monocular depth prediction on camera images. The method is evaluated on input recordings from the KITTI benchmark [11] [12], reporting a root mean square error of less than $1\ \frac{m}{s}$.

III. Data-driven Multiresolutional Learning for Accurate Parameter Estimation

Several data-oriented models proposed in various research works are relevant from a theoretical point of view for the problem of precise parameter estimation. In the field of human psychology and human memory research, Atkinson and Shiffrin [3] propose the dual-store model for the representation of human memory. According to this model, the human memory has a Sensory Register (SR) and two storage areas, the Short-Term Memory (STM) and the Long-Term Memory (LTM). Sensory information is first stored in the SR and, from there, transferred to the STM. The information stored in the STM decays and dissappears completely over time. Nevertheless, the information can be retained in the STM for a certain period of time via rehearsal mechanisms. Selected inputs from the LTM can also be transferred back to STM to serve as reference information for the recent inputs received from the SR. Losing et al. [2] propose the Self-Adjusting Memory (SAM) model for the k Nearest Neighbor (kNN) algorithm, which is partially based on the dual-store model in [3]. In the SAM architecture, current concepts stored in the STM and former concepts residing in the LTM are handled by dedicated models in accordance with the given situation.

An overview of our approach is depicted in Figure 1. Our concept is inspired by the dual-store model of the human memory, which comprises STM and LTM [3], and by the SAM architectural pattern [2]. The input datastream is processed and the situation reflected in the received data is evaluated. We are especially interested to determine whether any perturbations occur in the data and what is the magnitude of these perturbations. In case no disruptions are identified in the datastream or if these disruptions are smaller than a predefined threshold, we use long-term estimation mechanisms to approximate our parameters. We call this the *standard parameter value approximation*. The basis for our long-term estimation method are the system characteristics contained in the current system configuration. This is due to the fact that, in case of normal usage of the system, any changes to system characteristics are detectable over long periods of time, e.g., deterioration of vehicle tires.

However, in case of large disruptions in the datastream, we apply short-term estimation mechanisms in order to adjust the previously estimated standard value with a deviation error characteristic to the disruptive data. The basis for our short-term estimation approach are the input data received received by the system from its environment. Consider the example of a vehicle equipped with a GPS device. Disruptions in the GPS data may occur due to fluctuating GPS signal strength, which depends on the satellite geometry and on the road landscape.

 ISBN: 978-1-61208-771-9

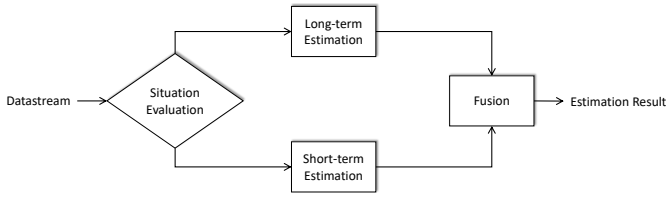

Figure 1. Overview of data-driven multiresolutiuonal learning.

In the current situation of the vehicle, old GPS data may not be relevant anymore. Therefore, recent GPS data must be taken into account in order to adjust the vehicle location displayed in the vehicle's navigation system.

Notice that the two categories of estimation mechanisms, short-term and long-term, work on differen time resolutions. As their names suggest, the short-term estimated values are updated more often than the long-term approximations. This is because system configurations change more slowly than current inputs from the system environment.

IV. Data-driven Vehicle Speed Estimation

In order to evaluate our concept, we build a case-study around an example system from the automotive domain. This section presents the system, which implements a vehicle speed estimation algorithm, together with the Euro NCAP requirements, as well as the preconditions and physical boundaries imposed on the system. The vehicle estimation algorithm has four major components: (A) estimation of the tire circumference, (B) plausibility check of the tire circumference, (C) computation and roundoff of the vehicle speed, and (D) smoothing of the vehicle speed curve (see Figure 2).

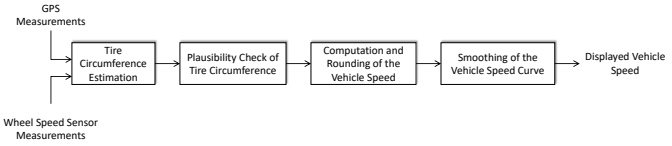

Figure 2. Overview of vehicle speed estimation algorithm.

A. System Requirements, Assumptions and Physical Boundaries

The proposed vehicle speed estimation algorithm must satisfy the NCAP requirement in (3):

$$0 \leq v_{display} - v_{real} \leq 5 \ \frac{km}{h} \tag{3}$$

where $v_{display}$ is the vehicle speed displayed on the dashboard of the car, and v_{real} is the actual vehicle speed. This requirement must be held under the assumption that the vehicle does not drive slower than $50 \ \frac{km}{h}$ or faster than $120 \ \frac{km}{h}$:

$$50 \leq v_{real} \leq 120 \ \frac{km}{h} \tag{4}$$

Notice that this requirement is stronger than the constraint imposed by the current european legal regulations [5].

The minimum and maximum values of the tire circumference are denoted TC_{min} and TC_{max}, and represent its lower and upper physical limits. Notice that these physical boundaries are specific for each tire profile. The vehicle speed estimation algorithm assumes the following boundaries for TC_{real}, the actual tire circumference:

$$TC_{min} = 2118 \ mm \ ; \ TC_{max} = 2293 \ mm$$

Based on the phyiscal boudaries of the tire circumference and on the maximum vehicle speed error of $5 \ \frac{km}{h}$ allowed by the NCAP requirement, we can derive the lower and upper physical limit, n_{min} and n_{max}, for the wheel speed $n_{current}$ measured by the wheel speed sensor:

$$n_{min} = 6.056 \ \frac{1}{s} \ ; \ n_{max} = 15.738 \ \frac{1}{s}$$

B. Estimation of the Tire Circumference

The estimated tire circumference has two components: (A) a system-oriented approximation of the standard tire circumference on the basis of wheel speed measurements and (B) an environment-oriented estimation of the tire circumference error on the basis of selected GPS data. Since system characteristics are subject to rather slow changes over time, long-term estimation is used to approximate the standard tire circumference. On the other hand, input data coming from the environment is subject to decay and becomes useless in a comparably short period of time. Thus, short-term estimation is more appropriate for the estimation of the tire circumference error. The computation of the estimated tire circumference is shown in (5):

$$TC_{learned} = TC_{standard} + \Delta TC_{learned} \tag{5}$$

Approximation of the Standard Tire Circumference. The standard tire circumference is estimated on the basis of the currently measured wheel speed $n_{current}$, according to the approximation curve defined by (6):

$$TC_{standard} = a \cdot sin(n_{current} - \pi) + b \cdot (n_{current} - 10)^2 + c \tag{6}$$

where $a = -0,5152$, $b = 0,07646$, and $c = 2175$. The coefficients a, b and c used in (6) have been chosen so that the curve matches approximately the ground truth of the tire circumference measurements. The ground truth data has been computed from the vehicle speed measurements performed with an ADMA sensor at a test facility of our industry project partner.

Estimation of the Tire Circumference Error. The tire circumference error is estimated on the basis of GPS data. The received GPS data contains GPS measurements of the tire circumference as well as information about the quality of the received data. Usually, a strong GPS signal means also a high quality of the received GPS data. Consequently, the error contained in high quality GPS data is very small. In order for the received GPS data to be considered for further computations, the error of the GPS data e_{GPS} must be under a certain threshold e_{max}^{GPS}. For our concept, we considered $e_{max}^{GPS} = 0.15$, which causes a deviation of the estimated vehicle speed of at most $\pm 0.15 \ \frac{m}{s}$. The main procedure for the estimation of the tire circumference error is described by Algorithm 1 depicted in Figure 3.

Notice that the computations in Algorithm 1 are controlled by a boolean flag, denoted as $updateFlag$. The update of the tire circumference error is performed only when certain conditions are met. These conditions are:

1) *small GPS data error:* $e_{max}^{GPS} = 0.15 \ \frac{m}{s}$,

 ISBN: 978-1-61208-771-9

Algorithm 1: Estimation of the tire circumference error.

procedure $TCError(TC^{GPS}_{measured}, n_{current}, e_{GPS}, a_{long}, a_{lat}, \alpha_{road})$

- $updateFlag \leftarrow UpdateFlag(e_{GPS}, a_{long}, a_{lat}, \alpha_{road})$
- $\Delta TC_{init} \leftarrow \frac{2.5}{n_{current}} \cdot \frac{1000}{3.6}$
- $\Delta TC_{max} \leftarrow \frac{5}{n_{current}} \cdot \frac{1000}{3.6}$
- $\Delta TC_{learned} \leftarrow \Delta TC_{init}$
- **if** $updateFlag \neq false$ **then**
  - $\Delta TC_{update} \leftarrow TCErrorUpdate(TC_{standard}, TC^{GPS}_{measured}, n_{current}, e_{GPS}, \Delta TC_{max})$
  - $\Delta TC_{learned} \leftarrow (1-w) \cdot \Delta TC_{learned} + w \cdot \Delta TC_{update}$
- **end if**
- **return** $\Delta TC_{learned}$

end procedure

Figure 3. Algorithm for the estimation of the tire circumference error.

2) *longitudinal acceleration limited by an upper bound:* $a^{long}_{max} = 0.001\ \frac{m}{s^2}$,
3) *small lateral acceleration to avoid skidding scenarios:* $a^{lat}_{max} = 0.0001 \frac{m}{s^2}$, and
4) *small road gradient:* $\alpha^{road}_{max} = 12\%$.

The computation of the update flag is depicted in Algorithm 2 (see Figure 4).

Algorithm 2: Computation of the update flag.

procedure $UpdateFlag(e_{GPS}, a_{long}, a_{lat}, \alpha_{road})$

- $gpsErrorFlag \leftarrow e_{GPS} \leq e^{GPS}_{max}$
- $roadSlopeFlag \leftarrow \alpha_{road} \leq \alpha^{road}_{max}$
- $longAccelFlag \leftarrow a_{long} \leq a^{long}_{max}$
- $latAccelFlag \leftarrow a_{lat} \leq a^{lat}_{max}$
- **if** ($gpsErrorFlag = false$ **or** $roadSlopeFlag = false$ **or** $longAccelFlag = false$ **or** $latAccelFlag = false$) **then**
  - **return** $false$
- **else**
  - **return** $true$
- **end if**

end procedure

Figure 4. Algorithm for the computation of the update flag.

Observe that in the first iteration of TCERROR procedure, the tire circumference error $\Delta TC_{learned}$ is initialised with an initial value ΔTC_{init}. Due to the NCAP requirement, the error $\Delta TC_{learned}$ has the upper bound ΔTC_{max}. Notice that the upper bound ΔTC_{max} necessarily depends on the maximum vehicle speed error permitted by the NCAP requirement.

In every subsequent iteration of the estimation algorithm TCERROR, the tire circumference error is updated on-the-fly based on current GPS measurements. Thus, the estimated tire circumference error is a function of previous estimations and updates based on current GPS data, as depicted in (7):

$$\Delta TC_{learned} = (1-w) \cdot \Delta TC_{learned} + w \cdot \Delta TC_{update} \quad (7)$$

where $w = 0.1$ is an application parameter.

In order to comply with the NCAP requirements, the updates to the tire circumference error are computed exclusively with adequate GPS data. Such data carries a maximum error of $e^{GPS}_{max} = 0.15$ and can be used for further computations. Any other received GPS data, which bears a larger error than the defined maximum threshold, is discarded. It is therefore necessary to define a mechanism by which intervals of good GPS data can be identified and selected from the entire GPS data batch received by the vehicle sensors. The mechannism for the selection of the GPS data is illustrated visually in Figure 5. Based on the selected GPS data, Algorithm 3 computes the update of the tire circumference error (see Figure 6).

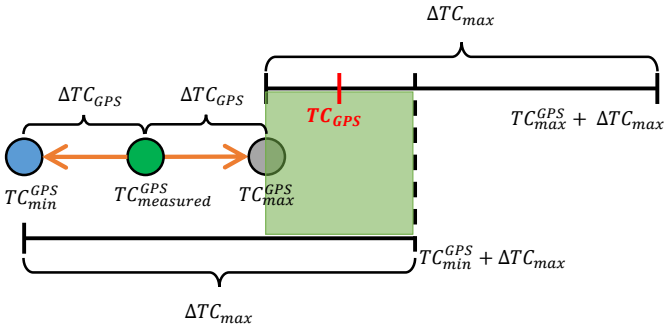

Figure 5. A visual intuition for the selection of adequate GPS data.

Algorithm 3: Computation of the tire circumference error update.

procedure $TCErrorUpdate(TC_{standatd}, TC^{GPS}_{measured}, n_{current}, e_{GPS}, \Delta TC_{max})$

- $\Delta TC_{GPS} \leftarrow \frac{e^{GPS}_{max} \cdot 1000}{n_{current}}$
- $TC^{GPS}_{max} \leftarrow TC^{GPS}_{measured} + \Delta TC_{GPS}$
- $TC^{GPS}_{min} \leftarrow TC^{GPS}_{measured} - \Delta TC_{GPS}$
- $TC_{GPS} \leftarrow \frac{TC^{GPS}_{max} + (TC^{GPS}_{min} + \Delta TC_{max})}{2}$
- $\Delta TC_{update} \leftarrow TC_{GPS} - TC_{standard}$

end procedure

Figure 6. Algorithm for the computation of the tire circumference error update.

The proposed algorithm takes further errors into consideration, e.g., errors due to the road gradient and rounding of the instrument display. We employ several mechanisms in order to compensate for these errors. Due to space restrictions, these mechanisms are not explained in this paper.

C. Plausibility Check of the Tire Circumference

For each tire profile, there are specific lower and upper limits, which constitute the physical boundaries of the real and of the estimated tire circumferences. Plausibility checks are necessary in order to eliminate any outliers.

However, before making any plausibility checks, we use a filter on the newly estimated tire circumference, in order to make sure that the difference between the new value and the old value estimated in the previous iteration does not exceed the threshold $P = 20\ mm$ in $3\ s$. An overshoot of the threshold P usually means that some of the data necessary for the estimation of the tire circumference has been missing, e.g., due to the vehicle not running. In this case, the old

 ISBN: 978-1-61208-771-9

data is not useful anymore, and therefore must be discarded. The computation continues only with the newly estimated tire circumference. Afterwards, the plausibility checks filter out the physically implausible values, i.e., values situated outside the interval spanned by the predefined physical boundaries $[TC_{min}, TC_{max}]$, as shown in (8):

$$TC_{plausible} = max(TC_{min}, min(TC_{learned}, TC_{max})) \quad (8)$$

D. Computation and Roundoff of the Vehicle Speed

The instanteneous vehicle speed is then computed based on the tire circumference and current wheel speed measured by the wheel speed sensors of the vehicle, as shown (9).

$$v = \frac{3.6}{1000} \cdot n_{current} \cdot TC_{plausible} \quad (9)$$

The speedometer dial in every vehicle displays a range of natural numbers from zero to an upper limit, which varies by make and model of the car. The displayed speed $v_{display}$ is computed by rounding off the vehicle speed v, so that it matches the numbers on the speedometer range.

E. Smoothing of the Vehicle Speed Curve

A smoothing function is applied to the resulting curve of the vehicle speed, in order to avoid the pointer needle of the speedometer bouncing back and forth at every small change in the estimation of the vehicle speed.

V. Experiments

We implemented the proposed vehicle speed estimation algorithm using MATLAB/SIMULINK and performed the evaluation on two driving scenarios. The data in both driving scenarios has been collected and provided by our industrial project partner, who collected the data using its own field test platform. The two scenarios are depicted in Figure 7 and in Figure 8 respectively, along with the algorithm evaluation. In both scenarios, the travelling time bears 1000 seconds, approximately 16.7 minutes. In each scenario, the first graph illustrates the evolution over time of three variables:

1) the 2D GPS speed measured with the GPS device of the test vehicle,
2) the actual vehicle speed, considered to be the ground truth and which is measured by ADMA sensors, and
3) the NCAP upper bound, which is the maximum speed allowed by the NCAP requirement.

The second graph shows the performance of our algorithm with respect to the maximum vehicle speed deviation permitted by the NCAP requirement.

The first scenario illustrates the ideal situation, specified also by the NCAP requirements and preconditions. The value range of the actual vehicle speed is situated between 50 $\frac{km}{h}$ and 130 $\frac{km}{h}$. The ADMA speed curve depicts a relatively smooth driving style, with clear-cut acceleration and decceleration segments and continuous periods of time with constant driving. It is fairly easy to see that in this scenario the vehicle speed deviation, $v_{display} - v_{real}$ is between cca 0.5 $\frac{km}{h}$ and cca 3.0 $\frac{km}{h}$, which satisfies the NCAP requirement specified in (3).

The second scenario depicts a more dynamic situation. The ADMA speed curve, with a value range between 0 $\frac{km}{h}$ and 180 $\frac{km}{h}$, illustrates a more sporty driving style, with abrupt speed-ups and sharp brakes, which alternate frequently. Notice that, after 100 s, the estimation of vehicle speed deviation

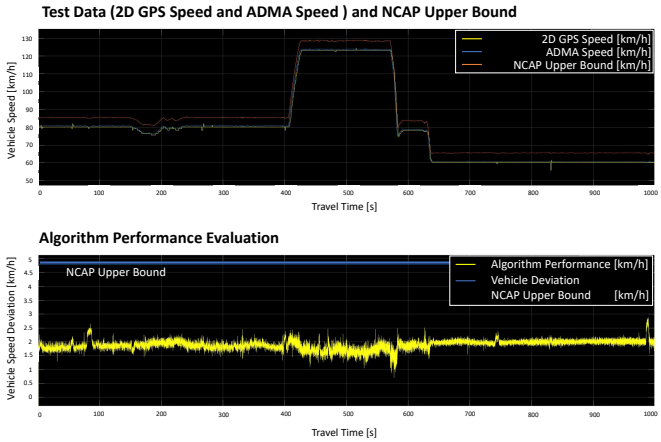

Figure 7. First scenario: smooth driving.

$v_{display} - v_{real}$ is stabilized between a minimum of cca 0.5 $\frac{km}{h}$ and a maximum of cca 3.5 $\frac{km}{h}$, thus satisfying the NCAP requirement.

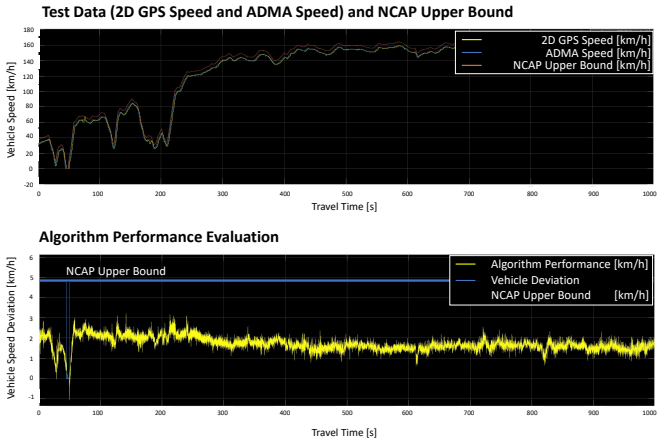

Figure 8. Second scenario: dynamic driving.

However, in this second scenario, the time period of the first 100 s is of particularly interest to us. Notice that the estimated vehicle speed deviation drops down to -1 $\frac{km}{h}$ around the middle of this time interval, meaning that the actual vehicle speed is underestimated. This occurrence can be attributed to the fact that, at that time, the ADMA speed decreases down to 0 $\frac{km}{h}$, i.e., the vehicle has stopped. It is obvious that no valid wheel speed measurements and GPS sensor data can be collected while the vehicle is stopped.

VI. Summary and Conclusions

We proposed an algorithm for the estimation of the ego-vehicle speed displayed on the dashboard instrument. Our approach is built upon the concepts of Short-Term Memory and Long-Termy Memory introduced by Atkinson and Shiffrin [3], which have been further developed in the Self-Adjusting Memory architectural pattern by Losing et al. [2]. We estimate the ego-vehicle speed by approximating its actual tire circumference.

 ISBN: 978-1-61208-771-9

In our approach, long-term estimation is used for the approximation of a standard tire circumference on the basis of current measurements of the wheel speed sensors. The wheel speed readings depend directly on the vehicle configuration, i.e., the profile of the tires mounted on the vehicle, which remains rather stable over time. An accurate approximation of the tire circumference is critical for a precise estimation of the displayed vehicle speed. Therefore, we use short-term estimation mechanisms to estimate a tire circumference error, with which the standard tire circumference is corrected. The short-term estimation method makes use of the sensor data collected with the ego-vehicle's GPS device, during the vehicle travel time. Not every received GPS data is used for the short-term estimation. Instead, we define a mechanism by which only the adequate data is selected and used for the estimation of the tire circumference error. We define several constraints, which specify in what sort of situations the short-term mechanism can be triggered. These criteria take into account the error of the GPS data, the vehicle's longitudinal and lateral acceleration and the road gradient. Through this approach, we are able to better control the process for the tire circumference approximation and, by extension, that of the vehicle speed estimation. Furthermore, our algorithm compensates for errors of the tire circumference estimation occurring due to the road gradient and the rounding off necessary for the speedometer dial. An experimental validation of the algorithm on two scenarios with real-world test data shows that the proposed approach performs well within the limits of the Euro NCAP requirements.

Nevertheless, there is potential for further optimization, which we intend to investigate in future research work. This optimization potential refers to the possible deviations, which may occur due to slippage, since slippage errors have a direct influence on the wheel speed measurements. For this, we need to perform an extensive analysis on a larger test data set. Moreover, it would be interesting to apply the presented vehicle speed estimation approach on test data gathered in more difficult driving conditions, e.g. patches of wet roadway alternating with dry road areas. Furthermore, we plan to extend our case study and investigate mechanisms for long-term estimation and short-term estimation, which can be used interchangeably in support of each other, i.e., use LTM to provide reference values for STM and STM to correct previously approximated values by LTM.

Since the speedometer is a critical vehicle instrument, experimental validation and testing are not enough to provide adequate confidence in the correct functioning of the vehicle speed estimation method. For this, we plan to use formal verification methods, since they are especially suitable to provide a mathematical proof that a system conforms to the legal and and the customer requirements. Furthermore, formal verification methods can be used to construct a solid argument for the system certification.

REFERENCES

[1] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of Probabilistic Real-time Systems," in Proc. 23rd International Conference on Computer Aided Verification (CAV'11), ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 585–591.

[2] V. Losing, B. Hammer, and H. Wersing, "Self-Adjusting Memory: How to Deal with Diverse Drift Types," in Proceedings of the 26^{th} International Joint Conference on Artificial Intelligence, IJCAI-17, 19-25 August 2017, Melbourne, Australia. IJCAI, Aug. 2017, pp. 4899–4903, Sierra, C. Ed., ISBN: 978-0-9992411-0-3, URL: https://www.ijcai.org/proceedings/2017/690 [accessed: 2020-01-07].

[3] R. C. Atkinson and R. M. Shiffrin, Human memory: A proposed system and its control processes. Academic Press, Jan. 1968, chapter 3, pp. 89–198, in Spence, K. W. and Spence, J. T. The Psychology of Learning and Motivation, ISBN: 0079-7421.

[4] W. Harris, "How Speedometers Work," 2007, URL: https://auto.howstuffworks.com/car-driving-safety/safety-regulatory-devices/speedometer.htm [accessed: 2020-01-09].

[5] "Richtlinie 75/443/EWG des Rates vom 26. Juni 1975 zur Angleichung der Rechtsvorschriften der Mitgliedstaaten über den Rückwärtsgang und das Geschwindigkeitsmeßgerät in Kraftfahrzeugen," 1975, URL: https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31975L0443:DE:HTML [accessed: 2020-01-06].

[6] K. Huang and C. Stachniss, "Joint Ego-motion Estimation Using a Laser Scanner and a Monocular Camera Through Relative Orientation Estimation and 1-DoF ICP," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 1-5 October 2018, Madrid, Spain. IEEE, Oct. 2018, pp. 671–677, Maciejewski, A. A. Ed., ISBN: 978-1-5386-8095-7, ISSN: 2153-0858, URL: https://doi.org/10.1109/IROS.2018.8593965 [accessed: 2020-01-08].

[7] S. Nedevschi, C. Golban, and C. Mitran, "Improving accuracy for Ego vehicle motion estimation using epipolar geometry," in 2009 12th International IEEE Conference on Intelligent Transportation Systems, 4-7 October 2009, St. Louis, MO, USA. IEEE, Oct. 2009, pp. 1–7, ISBN: 978-1-4244-5519-5 ISSN: 2153-0017, URL: https://doi.org/10.1109/ITSC.2009.5309610 [accessed: 2020-01-08].

[8] G. H. Lee, F. Fraundorfer, and M. Pollefeys, "Motion Estimation for Self-Driving Cars with a Generalized Camera," in 2013 IEEE Conference on Computer Vision and Pattern Recognition, 23-28 June 2013, Portland, OR, USA. IEEE, Jun. 2013, pp. 2746–2753, Kellenberger, P. Ed., ISSN: 1063-6919, URL: https://doi.org/10.1109/CVPR.2013.354 [accessed: 2020-01-08].

[9] X. Qimin, L. Xu, W. Mingming, L. Bin, and S. Xianghui, "A methodology of vehicle speed estimation based on optical flow," in Proceedings of 2014 IEEE International Conference on Service Operations and Logistics, and Informatics, 8-10 October 2014, Qingdao, China. IEEE, Oct. 2014, pp. 33–37, ISBN: 978-1-4799-6058-3, URL: https://doi.org/10.1109/SOLI.2014.6960689 [accessed: 2020-01-09].

[10] R.-A. Rill, "Speed estimation evaluation on the kitti benchmark based on motion and monocular depth information," 2019.

[11] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in Conference on Computer Vision and Pattern Recognition (CVPR), 2012.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," International Journal of Robotics Research (IJRR), 2013.

 ISBN: 978-1-61208-771-9