Periodica Polytechnica
Transportation Engineering

Design of Lane Keeping Algorithm of Autonomous Vehicle

44(1), pp. 60-68, 2016
DOI: 10.3311/PPtr.8177

Olivér Törő[1*], Tamás Bécsi[1], Szilárd Aradi[1]

RESEARCH ARTICLE

Received 26 April 2015; accepted after revision 14 September 2015

Abstract

The paper presents the design and realization of lane keeping function of an autonomous electric go-cart. The requirement towards the system concerning this paper is navigating the vehicle on a closed track with road markings, based on information from an optical camera with lane detection capabilities. To achieve this task, two solutions were used, a double-loop control with feedforward load disturbance compensation and a nonlinear method. The control algorithms were designed and tuned in a Hardware-In-The-Loop framework. The nonlinear algorithm was implemented on two different hardware devices and validated in CarSim–Matlab software environment.

Keywords

autonomous vehicle, lane keeping, line detection, hardware-in-the-loop

[1] Department of Control for Transportation and Vehicle Systems, Faculty of Transportation Engineering and Vehicle Engineering, Budapest University of Technology and Economics
H-1111 Budapest,Stoczek J. u. 2., Hungary

Olivér Törő, Researcher ID: J-2852-2015

Tamás Bécsi, Researcher ID: H-9818-2012

Szilárd Aradi, Researcher ID: H-9809-2012

[*] Corresponding author, e-mail: toro.oliver@mail.bme.hu

1 Introduction

Research on the subject of autonomous vehicles has been going on for decades. Although much progress in enhancing comfort and safety has been achieved, driverless cars for the public are not a reality yet.

There are different issues to be solved. On the technical side the lack of precise and at the same time inexpensive solutions for locating the vehicle position is a problem. Reliable operation of certain sensors can be ensured only in clean weather. There are sensors with performance beyond human senses, however evaluating the measurements is a hard task and far not as effective as vision based navigation of human drivers.

One of the most important legal obstacle is that vehicles without mentally and physically capable drivers are generally not permitted in public traffic.

With increasing automation level of vehicles participating in road traffic the number of human causalities shows decrease. Nowadays, as the number of drivers suffering fatal accidents dropped to a level comparable of pedestrians, research towards active and passive systems for protecting them came into focus.

Recent years basic Advanced Driver Assistance Systems (ADAS) functions such as Lane Keeping and Adaptive Cruise Control became common in lower middle class cars, making vehicle automation available to a wider public.

This paper concerns a university student project, which is carried out with the goal of developing an autonomous vehicle, capable of driving on a closed track. Navigation is based on a camera that can recognize the lane marker lines.

The topic of the present work is the design of lane keeping algorithm. In Section 2 the vehicle platform and the camera are presented in detail. The vehicle model and the theoretical background for control are covered in Section 3 and 4. For the controller design and validation, hardware-in-the-loop simulation was used. The setup and the results are presented in Section 5 and 6.

2 Vehicle platform

The vehicle is based on a common go-cart frame (see Fig. 1) with the internal combustion engine removed and a BLDC motor installed on it. The hydraulic disc brake and the trapezoidal

steering arm is also part of the frame. The steering rod is turned and positioned with a step motor. The closed-loop control for steering is achieved with a steering angle sensor, attached to the steering rod. The vehicle speed is calculated from the driving motor RPM, which is measured with an internal Hall sensor. The camera for lane detection is detailed in the subsequent section. For the control algorithms to run on, there are two possibilities. The Central Electric Control Unit, a 32 bit Atmel microcontroller based device and a National Instruments cRIO real time hardware module. The Actuator ECU, equipped with an 8 bit microcontroller is responsible for operating the actuators.

Additional information about the setup and the actuators can be found in Aradi et al. (2014).

Fig. 1 Autonomous test vehicle

2.1 Camera

The camera (Fig. 2) attached to the go-cart is a first generation Multi-Purpose Camera from Robert Bosch GmbH. The device, equipped with CMOS sensor, is commonly used in passenger cars involved in road traffic. The purpose of the camera is assisting the driver which means the following functions:

- Lane keeping support
- Lane departure warning
- Road sign recognition
- Object detection
- Intelligent headlight control

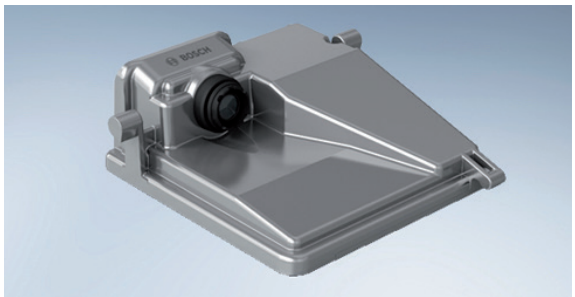

Fig. 2 Multi Purpose Camera (Robert Bosch GmbH)

The most important feature for our needs is line detection. Road surface markings, bitumen stripes or other line-like objects can be recognized and regarded as lane delimiters.

The software that runs on the microcontroller of the camera evaluates the image and transforms the detected lines to an appropriate coordinate system. The relative position of the vehicle to an identified line is described by two quantities:

- y: lateral error, the distance between the vehicle and the line
- Ψ: yaw error, the angle between the longitudinal axis of the vehicle and the tangent to the line

The reference frame (X_0,Y_0) for these quantities originates from a point on the delimiter line, designated by the perpendicular projection of the center of the vehicle's rear axle. The X_0 axis coincides with the tangent of the line at the origin, pointing to the direction of travel while the Y_0 axis is directed to the left (Fig. 3). As this frame moves along the delimiter line it will be called co-moving coordinate system.

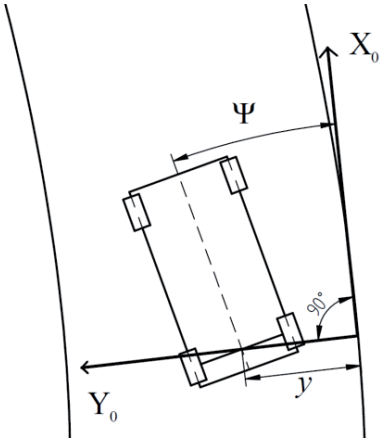

Fig. 3 Co-moving coordinate system for track errors

The theoretical road model used by the camera for curve fitting is the Euler spiral, which is a common track transition curve. The spiral cannot be handled conveniently by analytical methods, but since only the middle part of the whole spiral has practical usage, simple polynomial approximation can be used. The third-degree function

$$\frac{c_1}{6}x^3+\frac{c_0}{2}x^2, \tag{1}$$

interpreted in the co-moving coordinate system can equally be used for approximating transition curves or circular arcs and for describing straight lines. The coefficientsin the polynomial have direct meanings: c_0 represents the curvature while c_1 stand for the curvature gradient of the line at the origin.

As the camera is attached to the vehicle Eq. (1) modifies to

$$f(x) = \frac{c_1}{6}x^3 + \frac{c_0}{2}x^2 - \Psi x - y, \tag{2}$$

if written in the reference frame fixed to the vehicle.

The camera is capable of recognizing maximum four lines. Each line has its own coordinate system with the corresponding error quantities and properties such as color, existing probability and type (e.g. solid or dashed).

For the line detection to be effective the camera must be supplied with the following quantities describing the motion of the vehicle:

- yaw rate
- lateral acceleration
- longitudinal acceleration
- speed
- steering wheel angle
- steering wheel velocity

3 Vehicle model

For constructing the vehicle model, the following considerations should be taken. The suspensions of the wheels are rigid, therefore the roughness of the surface cannot be compensated and occasionally a wheel could lose contact with the ground. On the other hand, the rear axle is rigid, which produces additional unpredictable slip events during cornering. As these processes depend on independent factors they should be viewed as disturbanceand the consequent uncertainties should be compensated by the control.

Under these conditions using second order Newtonian dynamics would be cumbersome and unnecessary, hence kinematic bicycle model is chosen, which also means the reduction of the vehicle into a plane. Because the random slip events are already left out of the model, this reduction will not cause any unintended neglecting.

Difference from the regular formalism (Wang and Qi, 2001) is the neglect of the tire slip, so the wheels are only capable of moving along their longitudinal axis or turning, but no lateral motion is allowed.

The vehicle, as a rigid body performing planar motion, can be described with three coordinates. Considering a standing, inertial frame (Fig. 4), one is the yaw angle Ψ_v and the other two are the coordinates (x_v, y_v) of an arbitrary point of the go-cart. The natural choice is the rear axle because the speed v is measured at this point and the camera's line information is also computed relative to this reference. The speed of this point is considered to be the speed of the vehicle. A consequence of this choice is that the velocity of the vehicle is always parallel with the longitudinal axis and can be computed as follows:

$$\dot{x}_v = v\cos\Psi_v, \tag{3}$$

$$\dot{y}_v = v\sin\Psi_v. \tag{4}$$

The relation between the wheel angle δ and the path radius R is

$$\tan\delta = \frac{L}{R}, \tag{5}$$

where L is the wheelbase.

The yaw rate of the vehicle is

$$\dot{\Psi}_v = \frac{v}{R}, \tag{6}$$

combined with (5) gives

$$\dot{\Psi}_v = \tan\delta\frac{v}{L}. \tag{7}$$

With the approximation $\tan\delta \approx \delta$ the mistake taken is 1–2% in the realizable interval, however we can use linear formulae.

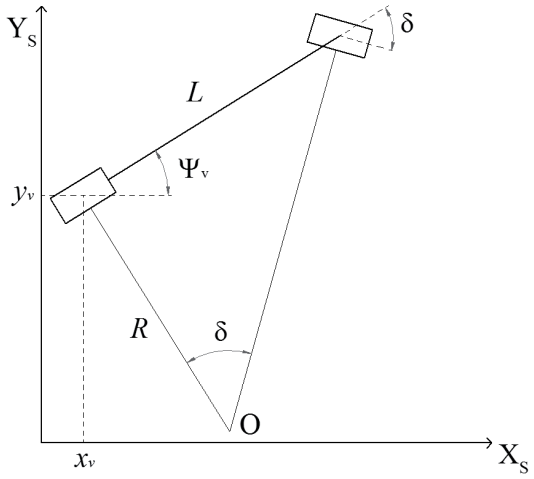

Fig. 4 Bicycle model depicted in the standing reference frame

4 Control

Longitudinal speed control is managed independently of the lateral motion and achieved with PI controller, based on RPM measurement.

With lateral control the goal is to keep the vehicle on the desired path and minimize the track errors. Two control strategies, a linear and a nonlinear one has been implemented.

4.1 Linear control

To design the control, the vehicle model has to be transformed from the standing reference frame (X_S,Y_S) to the co-moving (X_0,Y_0), where the track error measurements are taken. The origin of the latter frame has the coordinates (x_c, y_c) and yaw angle Ψ_c in the standing frame (Fig. 5). For the vehicle position vector components the transformation consists of a shift and a rotation:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\Psi_c & \sin\Psi_c \\ -\sin\Psi_c & \cos\Psi_c \end{bmatrix} \begin{bmatrix} x_v - x_c \\ y_v - y_c \end{bmatrix}. \tag{8}$$

The velocity components in the co-moving frame are:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} \cos\Psi_c & \sin\Psi_c \\ -\sin\Psi_c & \cos\Psi_c \end{bmatrix} \begin{bmatrix} \dot{x}_v - \dot{x}_c \\ \dot{y}_v - \dot{y}_c \end{bmatrix}, \tag{9}$$

where

$$\dot{x}_c = v\cos\Psi_c, \tag{10}$$

and

$$\dot{y}_c = v\sin\Psi_c, \tag{11}$$

After performing the multiplications and sorting out, the second component of (9) reads

$$\dot{y} = v\sin\Psi, \tag{12}$$

where

$$\Psi = \Psi_v - \Psi_c \tag{13}$$

is the yaw error. By the nature of the co-moving coordinate system, x and $\dot{x}$ are always zero and y stands for the lateral error. The yaw error rate is given by the difference

$$\dot{\Psi} = \dot{\Psi}_v - \dot{\Psi}_c, \tag{14}$$

where

$$\dot{\Psi}_c = v\varrho. \tag{15}$$

The road curvature ϱ is understood at the origin and has the same sign convention as the yaw error.

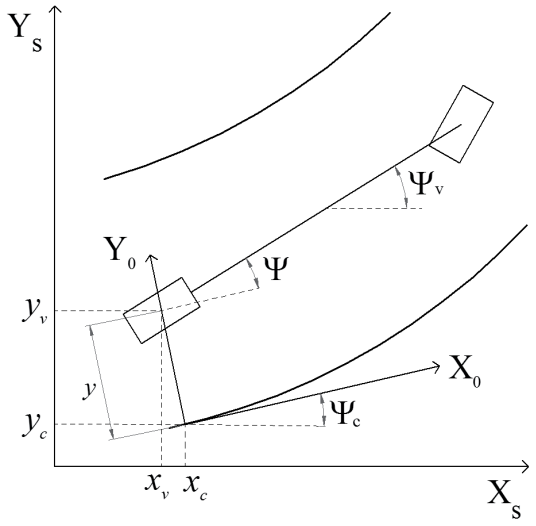

Fig. 5 The vehicle model and the co-moving coordinate system in the standing frame

Introducing the error state $e = [y, \Psi]$ Eqs. (12) and (14) in state space representation takes the form

$$\dot{e} = Ae + B_1\delta + B_2\varrho, \tag{16}$$

where $B_1 = [0, v/L]$, $B_2 = [0, -v]$, and the system matrix is

$$A = \begin{bmatrix} 0 & v \\ 0 & 0 \end{bmatrix}. \tag{17}$$

Beside the approximation $\tan\delta \approx \delta$, $\sin\Psi \approx \Psi$, is also used.

While stabilizing the system (A, B_1) with state feedback and adding ϱ as a load in the form

$$\dot{e} = (A - B_1K)e + B_2\varrho \tag{18}$$

is an option (Rajamani, 2006), there are other possibilities. The road curvature acts as a disturbanceon the system, but its value is known thus its effect can be cancelled with a feedforward term (Moon and Choi, 2011).

The scheme is the following. Using s-Domain notations, consider a plant $P(s)$ with its output disturbed with the signal $N(s)$ through a transfer function $G_N(s)$ (Fig. 6). The feed forward compensator can be constructed as

$$C_N(s) = -\frac{G_N(s)}{P(s)}. \tag{19}$$

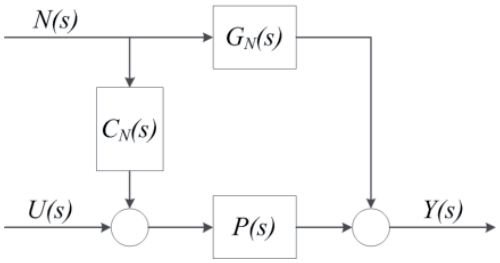

Fig. 6 Scheme of feedforward disturbance cancellation

Taking the lateral error as output and the wheel angle as input the transfer function of system (16) is

$$G(s) = \frac{v^2}{Ls^2}, \tag{20}$$

Note, that because of the inputs of (16) only have direct effects on the yaw error, $G(s)$ can be written as $G_1(s)G_2(s)$, where $G_1(s)=v/s$ and $G_2(s)=v/(Ls)$. Decoupling $G(s)$ into a product allows the application of a double-loop feedback control logic (Marino et al., 2011) with the following benefits. The yaw error in the inner loop can be compensated immediately before it adds up to a significant lateral error. On the other hand, in this structure we have a direct control over the yaw error state. For generating the control signal and reference yaw angle error from the lateral error, P and PD controllers are used (Fig. 7). Since the system is of integral type, reference tracking can be realized without the I term.

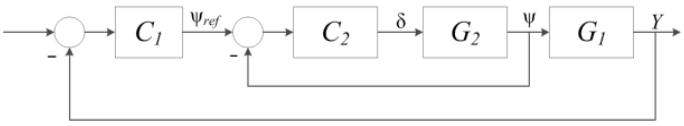

Fig. 7 Double loop control

Combining the double loop control with the feedforward compensator we get the final structure presented in Fig. 8. According to Eq. (19) the compensator is

$$C_N(s) = -\frac{G_N(s)}{G_2(s)}. \tag{21}$$

From Eq. (16), taking ϱ as input and Ψ as output we get $G_N = -v/s$, which, with (23) leads

$$C_N(s) = L \tag{22}$$

for the compensator.

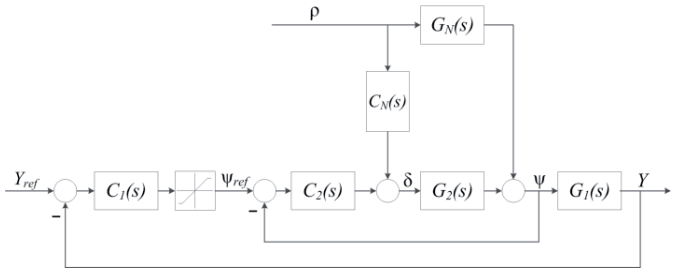

Fig. 8 Double loop control with feed forward compensator

By saturating Ψ_{ref}, a safety constraint can be made for preventing the yaw error to exceed a certain level.

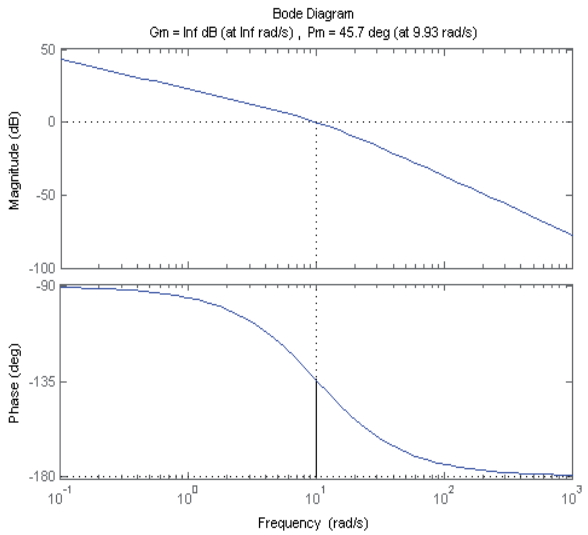

Fig. 9 Bode diagram of the controlled system

Tuning the controllers yielded a system with 45.7° phase margin (Fig. 9). For C_2, the proportional gain is $K_{P2} = 2.2$ and for $C_1 K_{P1} = 0.64$ with the derivative gain $K_{D1} = 0{,}09$.

Using look ahead distance d, the feedback signal for C_1 changes to $y + d\Psi$. With $d = 2$m, to ensure stability, the gains had to be lowered to $K_{P2} = 1.8$ and $K_{D1} = 0.03$. With this setup the achieved phase margin is 57.6°.

4.2 Nonlinear control

Another approach for lane keeping is the Stanley method which is a nonlinear control used by Stanford University's autonomous vehicle at the DARPA Grand Challenge (Thrun et al., 2006). The bicycle model used for this control differs from the one detailed in Section 3 in the chosen point describing the motion of the go-cart (Fig. 10). Using the center of the first axle as reference point, new quantities are introduced:

- v_{st}: vehicle speed, computed at the front axle. Its direction is parallel to the front wheel,
- y_{st}: lateral error at the front axle,
- Ψ_{st}: yaw error at the front axle.

The yaw error at the front and the rear axle differs in the angle the road bends over the L distance. As the curvature of the road is small, this difference can be neglected and Ψ will be used for the Stanley controller too. Similarly, the curvature of the road changes little, therefore $\dot{\Psi}$ will be used for the yaw error rate at the front axle:

$$\dot{\Psi} = v\frac{\tan\delta}{L} = \frac{v_{st}}{R_{st}} = v_{st}\frac{\sin\delta}{L}. \tag{23}$$

The relation between the speed at the front and rear axle is:

$$v_{st} = \frac{v}{\cos\delta}. \tag{24}$$

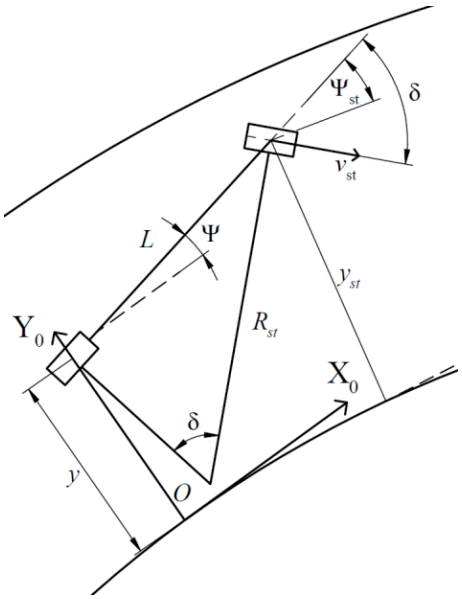

Fig. 10 Vehicle model for the Stanley-method

For computing the lateral error y_{st} from the measured y the orientation of the vehicle should be considered:

$$y_{st} = y + \Psi L. \tag{25}$$

Again, the effect of road curvature is neglected.

Finally, the dynamics of the lateral error at the front axle is given by

$$\dot{y}_{\mathrm{st}} = v_{\mathrm{st}} \sin(\Psi + \delta). \tag{26}$$

The input for the system can be constructed with the nonlinear function (Thrun et al., 2006):

$$\delta = -\left(\Psi + \arctan\left(k \frac{y}{v} \right) \right), \tag{27}$$

where k is the gain factor. The value for δ should be saturated to the realizable interval ($\delta_{\min}$, $\delta_{\max}$).

Analysis shows (Hoffmann et al., 2007), that the origin is the only equilibrium point in the (y_{st}, Ψ) phase space, and the system is globally asymptotically stable with linear or exponential convergence.

The phase portrait of the system, with gain parameter k = 3 s^{-1}, v =50 km/h speed and saturation (-20°,20°) is shown in Fig. 11. Above 3.3s^{-1} gain, the attractor starts to curve and crosses the axes multiple times, which could lead to oscillations.

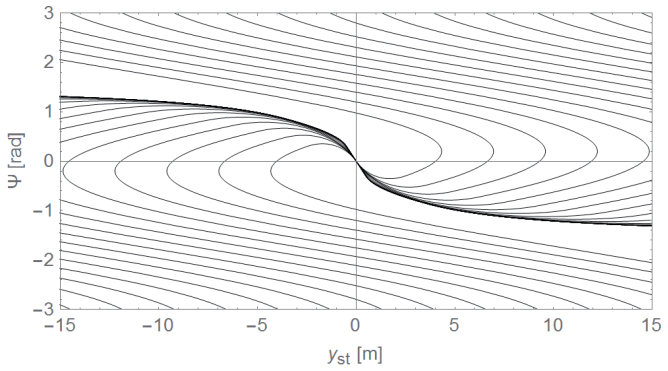

Fig. 11 Phase portrait of the system using the Stanley controller

Advantages of this method are easy implementation and low computational cost.

5 Hardware-in-the-loop framework

For tuning and validating the control Hardware-in-the-Loop (HIL) framework was used.

The automotive industry, where "real world" testing is resource consuming and often can be dangerous, is a typical user of HIL (Yan et al., 2002). A vehicle driven by a human driver involves great expenses and risks, moreover test reproducibility is limited. The driver, the vehicle or even both can be replaced by a model, which allows shorter testing cycles, lowers costs and increases reproducibility (Gietelink et al., 2006; Wehner et al., 2014). Sensors can also be tested efficiently in HIL (Coşkun et al., 2010). As real damage cannot be taken, test conditions are not limited by risk factors. Beside vehicle development, human driver model plays a key role in traffic management. (Dresner and Stone, 2004; Basak et al., 2013).

Work with the camera started in a simple HIL system. Standing before a monitor the camera was provided a video footage for calibration purposes. The video was recorded with a similar camera in a test car, involved in the real traffic situation. With the help of a field of view adjusting lens, reliable operation of the camera was guaranteed.

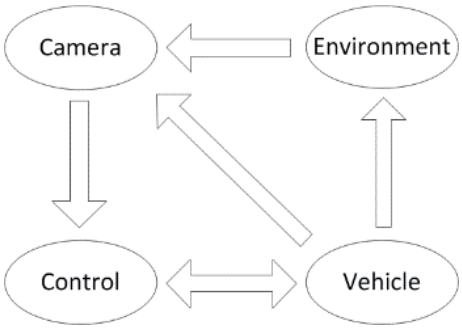

Fig. 12 Control scheme of the autonomous vehicle

Now turning to the vehicle platform, the control scheme with the involved hardware components can be seen in Fig. 12. The camera observes the environment, takes into account the state of the vehicle and provides the line information for the control. The control algorithm, also using data regarding vehicle kinematics, computes the reference signal for the actuators. The vehicle behaves according to the control and consequently the environment seen by the camera changes. Arrows indicate the direction of information flow.

For developing and testing purposes, the simulation of the vehicle and the environment was performed coherently in CarSim, a software for computing and visualizing vehicle-environment interactions and dynamics. For executing the control algorithm Simulink was used. The setup layout can be seen in Fig. 13. As the camera is communicating via CAN, a hardware interface was installed between Simulink and the sensor. For exchanging information with CarSim, a proper Simulink block was used.

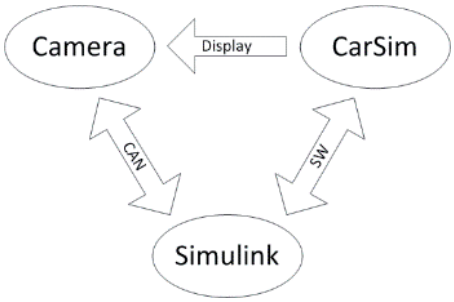

Fig. 13 Hardware-in-the-loop setup

With the friction coefficient on the road set to μ = 1 the upper limit for the desired speed can be estimated as

$$v = \sqrt{\mu g R_{\min}}, \tag{28}$$

where the radius of the sharpest turn on the track (Fig. 14) is $R_{\min}$ =30 m. With constant v = 50 km/h speed the vehicle can perform cornering without a considerable slip.

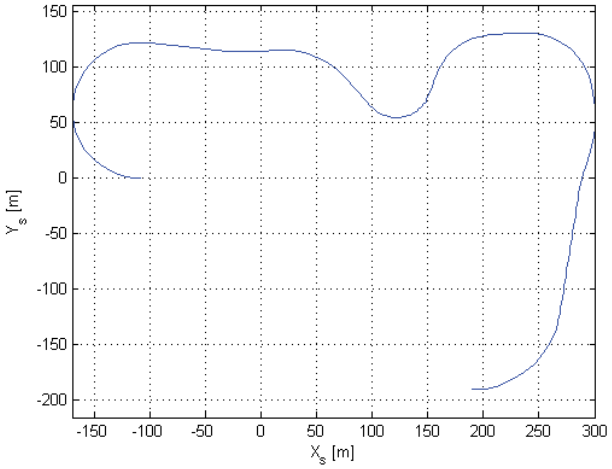

Fig. 14 The track used for the simulations, depicted in the standing coordinate system

5.1 Simulating the camera

The simulation was carried out in Simulink. Only the required functions were implemented, namely calculating the fitted curve and producing the appropriate CAN messages. Line detection was not an issue since the coordinates of the track are available from CarSim.

CarSim generates the smooth road from a set of holder points, about 6.5m distance from each other, with cubic spline interpolation. Only the holder points of the lane centerline are available to us, so the curve fitting is based on these points. The coordinates (x_h, y_h) of the holder points are in a global reference frame, but the camera works in a frame attached to the vehicle. For that reason to get the road coordinates relative to the camera the transformation

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\Psi_v & \sin\Psi_v \\ -\sin\Psi_v & \cos\Psi_v \end{bmatrix} \begin{bmatrix} x_h - x_v \\ y_h - y_v \end{bmatrix} \tag{29}$$

has to be applied. The yaw angle can also be acquired from CarSim. The function to be fit is the one described in Eq. (2):

$$f(x) = ax^3 + bx^2 + cx + d. \tag{30}$$

Using least squares method the sum to be minimized is

$$S = \sum_{i=1}^{N} (ax_i^3 + bx_i^2 + cx_i + d - y_i)^2, \tag{31}$$

where x_i and y_i are the coordinates from (31). Keeping in mind that the separation between the points is fixed, N defines the distance the fit is based on. Through the whole simulation $N = 8$ was used.

Figure 15 shows the quality of the fit regarding parameter b, that is twice the curvature. The dotted line represents the curvature of the road exported from CarSim, whereas the solid line stands for the estimated value. The saw tooth-like feature that can be seen in the zoomed area is caused by the entering of a new holder point after every 6.5m travel. The small steps indicate the cycle time of the fit, which is 0.1s.

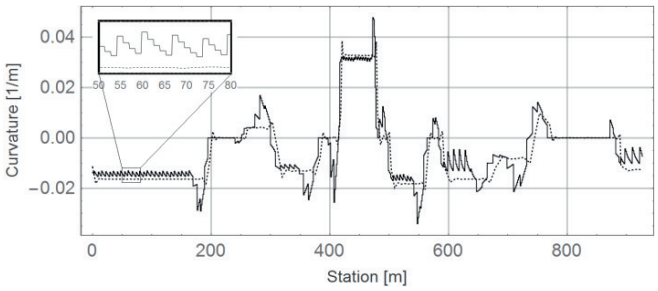

Fig. 15 Road curvature (dotted line)and the estimated value (solid line)

The control algorithms need a lateral offset value to track. Multiple lines may exist in certain situations, however a reference trajectory has to be created from them. Although more than one recognized line can increase reliability, for our purpose a single line is enough because the reliability of computer vision is not a problem in the simulation.

The reference trajectory is chosen to be the center line of the lane, thus only one curve fit is performed in each time step.

6 Results

The obtained parameters a,b,c,d stand for the estimation of the variables $c_1/6$, $c_0/2$, $-\Psi$, $-y$, respectively. These can be used as input to the controllers in Simulink.

For comparison, the performance of both controllers with track errors directly from CarSim can be seen in Fig. 16 and Fig. 17. The Stanley controller clearly outperforms the linear one, with fewer transients, overshoots and smaller steady state errors in curves.

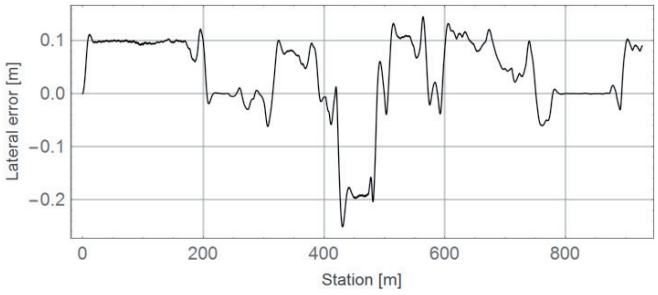

Fig. 16 Lateral error with linear controller – direct feed from CarSim

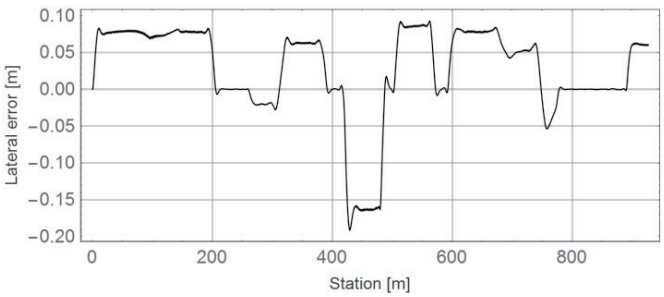

Fig. 17 Lateral error with Stanley controller – direct feed from CarSim

Figure 18 and Fig. 19 shows track errors realized with the linear controller using the simulated line data. The uncertainties, observed in the fitted curvature data causes oscillations in the lateral and yaw error too.

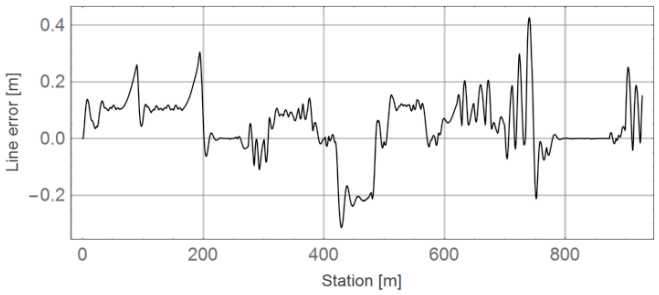

Fig. 18 Lateral error with linear controller

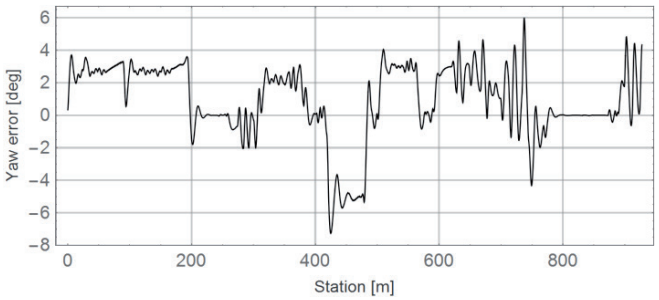

Fig. 19 Yaw error with linear controller

Using 2m look-ahead distance, we get more precise tracking and fewer oscillations (Fig. 20 and 21).

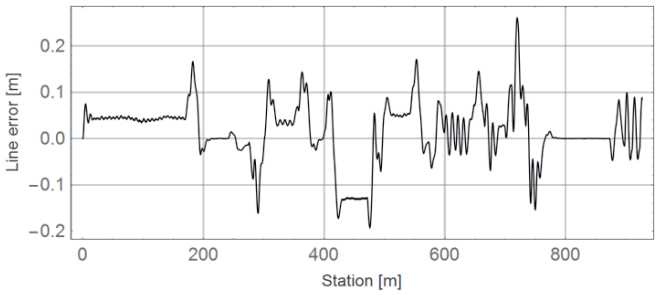

Fig. 20 Lateral error with linear controller using look ahead

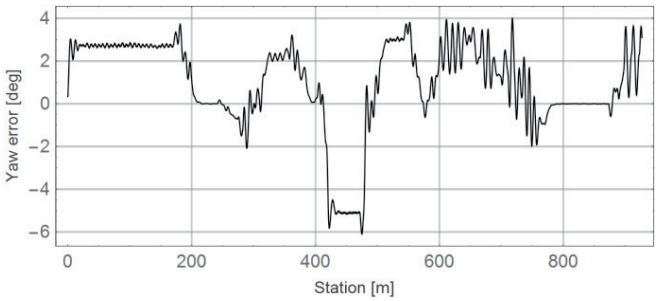

Fig. 21 Yaw error with linear conroller using look ahead

The nonlinear controller does not use a look-ahead distance. Its performance (Fig. 22 and 23) is comparable to the linear controller with the look-ahead but shows less oscillation.

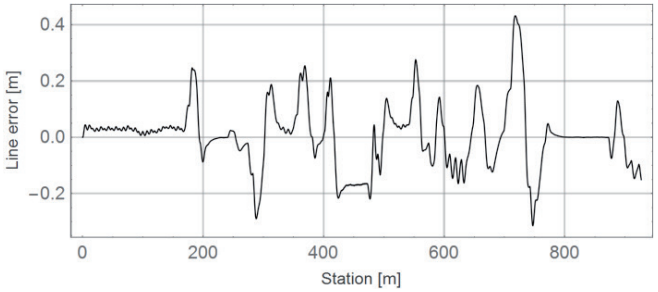

Fig. 22 Lateral error with Stanley controller

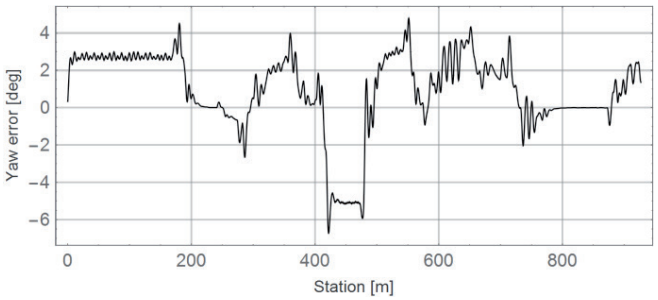

Fig. 23 Yaw error with Stanley controller

Although the theoretical performance of the nonlinear controller is better, it is more sensitive to the uncertainties of parameter estimation and produces more overshoots.

The Stanley method has been implemented on cRIO and on the Central ECU. Figure 24 shows the experimental setup for testing purposes.

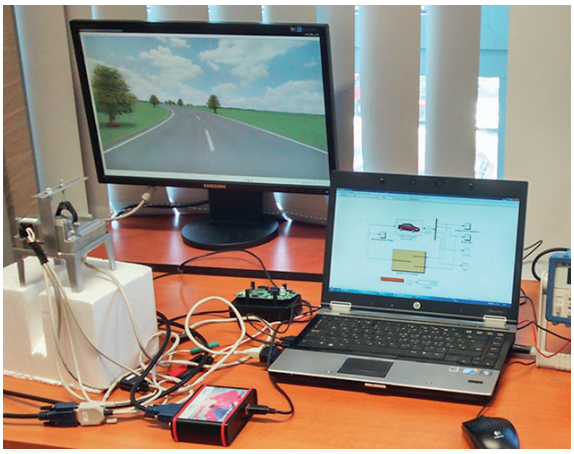

Fig. 24 Testing the control implemented on the Central ECU

Validation was carried out in the configuration shown in Fig. 25.

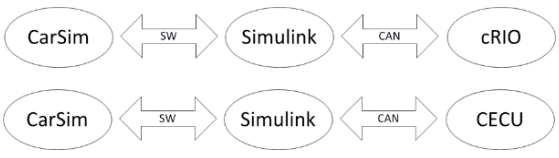

Fig. 25 Scheme for validating the implemented control algorithm

Exchanging the software controller of the HIL system with the implemented controls on CECU and the cRIO show the same performance as presented above.

6 Conclusion

Two lane keeping algorithms were presented, a linear and a nonlinear one. With comparable performances, the computation costs and advantages are different. Reliable operation in a real environment has not been achieved yet, however prior tests indicate that the camera is sensitive to shocks and line detection will not work when the yaw angle exceeds a certain level. Should keeping the yaw angle in an optimal interval be a problem the double loop control has to be implemented, as it allows direct control over that state.

Beside lane keeping, the vehicle could be enhanced with other functions, such as object avoidance and automatic parking. For these capabilities trajectory tracking is required, which also offers other control methods for lane keeping. Since lane detection involves looking ahead and the estimated track ahead the vehicle is available, model predictive control is also an option.

References

Aradi, S., Becsi, T., Gaspar, P. (2014) Experimental vehicle development for testing autonomous vehicle functions. In: *10th International Conference on Mechatronic and Embedded Systems and Applications*. Senigallia. Sept 10-12. DOI:10.1109/MESA.2014.6935534

Basak, K., Hetu, S., Li, Z., Azevedo, C. L. (2013) Modeling reaction time within a traffic simulation model. In: *Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems*. The Hague. Oct. 6-9. pp. 302-309. DOI: 10.1109/ITSC.2013.6728249

Baskar, L. D., Schutter, B., Hellendoorn, J., Papp, Z. (2011) Traffic control and intelligent vehicle highway systems: a survey. *IET Intelligent Transport Systems*. 5 (1). pp. 38-52. DOI: 10.1049/iet-its.2009.0001

Coşkun, F., Tunçer, O., Karsligil, M. E., Guvenc, L. (2010) Real time lane detection and tracking system evaluated in a hardware-in-the-loop simulator. In: *13th International IEEE Conference on Intelligent Transportation Systems*. Funchal. Sept. 19-22. pp. 1336-1343. DOI: 10.1109/ITSC.2010.5625111

Dresner, K., Stone, P. (2004). Multiagent Traffic Management: A Reservation-Based Intersection Control Mechanism. In: *The Third International Joint Conference on Autonomous Agents and Multiagent Systems*. New York. July 23. pp. 530-537. DOI: 10.1109/AAMAS.2004.190

Gietelink, O., Ploeg, J., Schutter, B., Verhaegen, M. (2006) Development of advanced driver assistance systems with vehicle hardware-in-the-loop simulations. *Vehicle System Dynamics*. 44 (7). pp. 569-590. DOI: 10.1080/00423110600563338

Hoffmann, G. M., Tomlin, C. J., Montemerlo, M., Thrun, S. (2007) Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing. In: *2007 American Control Conference*. New York. July 9-13. pp. 2296-2301. DOI: 10.1109/ACC.2007.4282788

Marino, R., Scalzi, S., Orlando, G., Netto, M. (2011) Nested PID steering control for lane keeping in autonomous vehicles. *Control Engineering Practice*. 19 (12). pp. 1459–1467. DOI: 10.1016/j.conengprac.2011.08.005

Moon, C., Choi, S. B. (2011) A driver model for vehicle lateral dynamics. *International Journal of Vehicle Design*. 56 (1-4). pp. 49-80. DOI: 10.1504/IJVD.2011.043258

Rajamani, R. (2006) *Vehicle Dynamics and Control*. New York: Springer.

Robert Bosch GmbH (2015) „Multi Purpose Camera." [Online] Available from: http://www.bosch-mobility-solutions.us/en_us/us/component_us/SF_PC_DA_Lane-Assist_SF_PC_Driver-Assistance-Systems_586.html?compId=72 [Accessed 15th February 2015]

Thrun, S. et al. (2006) Stanley: The Robot that Won the DARPA Grand Challenge. *Journal of Field Robotics*. 23 (9). pp. 661-692. DOI: 10.1002/rob.20147

Wang, D., Qi, F. (2001) Trajectory planning for a four-wheel-steering vehicle. In: *IEEE International Conference on Robotics and Automation*. Seoul. May 20-26. pp. 3320-3325. DOI: 10.1109/ROBOT.2001.933130

Wehner, P., Schwiegelshoh, F., Gohringer, D., Hubner, M. (2014) Development of driver assistance systems using virtual hardware-in-the-loop. In: *14th International Symposium on Integrated Circuits*. Singapore. Dec. 10-12. pp. 380-383. DOI: 10.1109/ISICIR.2014.7029542

Yan, Q., Williams, J., Li, J. (2002) Chassis Control System Development Using Simulation: Software in the Loop, Rapid Prototyping, and Hardware. In: *Loop, Proceedings of the 2002 SAE Automotive Dynamics & Stability Conference and Exhibition*. Detroit. May 7-9. pp. 337. DOI: 10.4271/2002-01-1565