2018-01-0608 Published 03 Apr 2018

Camera Based Automated Lane Keeping Application Complemented by GPS Localization Based Path Following

Mustafa Ridvan Cantas and Levent Guvenc The Ohio State University

Citation: Cantas, M.R. and Guvenc, L., "Camera Based Automated Lane Keeping Application Complemented by GPS Localization Based Path Following," SAE Technical Paper 2018-01-0608, 2018, doi:10.4271/2018-01-0608.

Abstract

Advances in sensor solutions in the automotive sector make it possible to develop better ADAS and autonomous driving functions. One of the main tasks of highway chauffeur and highway pilot automated driving systems is to keep the vehicle between the lane lines while driving on a pre-defined route. This task can be achieved by using camera and/or GPS to localize the vehicle between the lane lines. However, both sensors have shortcomings in certain scenarios. While the camera does not work when there are no lane lines to be detected, an RTK GPS can localize the vehicle accurately. On the other hand, GPS requires at least 3 satellite connections to be able to localize the vehicle and more satellite connections and real-time over-the-air corrections for lane-level positioning accuracy. If GPS localization fails or is not accurate enough, lane line information from the camera can be used as a backup. In this paper, a vision based lane keeping system is aided by a GPS based path following application to overcome the shortcomings of the GPS and camera sensors when used alone in highway driving path following applications. The developed system has a parameter space based robust steering controller which can handle lateral motion control of the vehicle based on path tracking error detected using the GPS or camera sensor. The designed control system works for both low speed and high-speed driving scenarios and is robust to changes in vehicle mass. The results are demonstrated using the validated model of our 2017 Ford Fusion Hybrid research automated driving vehicle in our hardware-in-the-loop simulator. Experimental verification is also planned.

Introduction

There are six automated driving levels defined in new SAE International standard J3016 varying from 0 to 5 [1], where 0 represents the no automation case and 5 represents the fully autonomous driving case with no human intervention. This paper will cover a lane keeping application for a vehicle already equipped with an adaptive cruise control. This automation level falls within Level 2 which is partial automation where the steering and acceleration of the vehicle are handled by the automated driving system but the driver is still in the loop. This is an initial part of our work aimed at developing a Level 3 Highway Chauffeur and a Level 4 Highway Autopilot in a realistic hardware-in-the-loop simulation (HIL) environment.

In the literature and in production level vehicles, there are many lane-keeping and lane departure warning applications. For instance, Tuncer et al. worked on developing a lane keeping system when the driver is inattentive [2]. In their application, a camera based lane keeping controller is designed and simulated in the HIL simulator. Kang et al. proposed a solution for estimating the lane positions for short term lane information lost from the camera [3]. Although lane keeping applications and path fallowing applications are thoroughly studied in the literature, the failure of the existing systems would not be acceptable for a fully autonomous vehicle system. Considering many of these systems are using the camera to detect the lane lines and localize the vehicle in the lateral direction, the failure of the camera detection would result in failure to keep being within the lane. As highlighted in the work of the Yenikaya et al. [4], some of the camera detection failures can be caused by the absence of the lane lines, poor lane line quality, shadow on the lane lines, or other vehicle occlusions. The camera may also completely fail to work or communicate with the controller. Today high accuracy GPS units are also available for accurate localization. For example, the GPS unit used in our experimental vehicle OXTS xNAV550 has 1.6 m accuracy with single antenna, 0.4 m for DGPS mode and up to 2 cm for RTK mode using a base station. Also with the use of online RTK correction services and RTK Bridge units it is possible to have RTK corrections without a base station. In the case of using RTK bridge unit, accuracy of the system is around 5 cm.While today RTK GPS units are very expensive as compared to the cameras, they are getting cheaper with the advance of the technology. Therefore, usage of a GPS based lane level path fallowing algorithm is suggested as one of the backup solutions for the camera failure cases. One might ask why GPS system is not used solely for the lane keeping application. This is because the GPS system also has its own shortcomings. If the RTK corrections for the GPS are not available or the lane level map of the environment is not

available, it is not possible to localize the vehicle within lane level accuracy. Therefore, a combination of the camera and GPS solution is preferred over using them alone by themselves.

The rest of the paper has sections in the following order: Lateral vehicle model, lane detection, path generation, lateral deviation calculation, lateral controller design, simulation results and summary/conclusions.

Lateral Vehicle Model

In this paper, lateral dynamics of the vehicle is modeled using the nonlinear vehicle model (Bicycle Model). In this model, the two front wheels are represented as single front wheel and similarly, the two rear wheels of the vehicle are represented as a single rear wheel. As our test vehicle is only steerable from the front wheels, the test vehicle is modeled to be only steerable from the front wheel [5]. Forces acting on the vehicle in this model are shown in Figure 1. Lateral forces generated by the front/rear wheels, vehicle center of gravity, distance of the center of gravity from the wheels and the preview distance are represented in the figure as F_f/ F_r, CG, l_f/ l_r, l_s respectively.

The lateral direction steering controller for the automated lane-keeping application is designed using a linearized version of the nonlinear vehicle model. Linearized state space model of the lateral motion of the vehicle is given in Equation 1 where β is the vehicle side slip angle at the vehicle center of gravity, r is vehicle yaw rate, V is velocity, $\Delta\psi$ is yaw angle of the vehicle with respect to desired path's tangent, ρ_{ref} is the road curvature, δ_f is the steering wheel angle and μ is the friction coefficient of the road. The entries a_{11}, a_{12}, a_{21}, a_{22}, b_{11}, b_{12} used in Equation 1 are given in Equations 2-7, where c_r, c_f are the cornering stiffness of the rear and front wheels, $\tilde{J} = J/\mu$ is the virtual mass moment of inertia and the $\tilde{m} = m/\mu$ is the virtual mass.

$$\begin{bmatrix} \dot{\beta} \\ \dot{r} \\ \Delta\dot{\psi} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ V & l_s & V & 0 \end{bmatrix} \begin{bmatrix} \beta \\ r \\ \Delta\psi \\ y \end{bmatrix} + \begin{bmatrix} b_{11} & 0 \\ b_{21} & 0 \\ 0 & -V \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_f \\ \rho_{ref} \end{bmatrix} \quad (1)$$

$$a_{11} = \frac{-(c_r + c_f)}{\tilde{m}V}, a_{12} = -1 + \frac{-(c_r l_r - c_f l_f)}{\tilde{m}V^2}, \quad (2)$$

$$a_{11} = \frac{-(c_r + c_f)}{\tilde{m}V}, a_{12} = -1 + \frac{-(c_r l_r - c_f l_f)}{\tilde{m}V^2}, \quad (3)$$

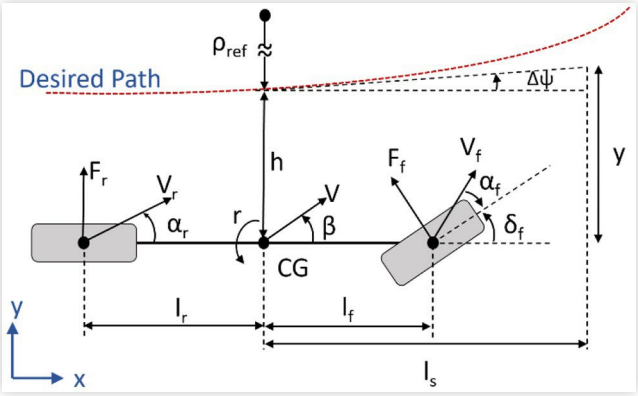
$$a_{21} = \frac{(c_r l_r - c_f l_f)}{\tilde{J}}, a_{22} = \frac{-(c_r l_r^2 + c_f l_f^2)}{\tilde{J}V^2}, \quad (4)$$

$$a_{21} = \frac{(c_r l_r - c_f l_f)}{\tilde{J}}, a_{22} = \frac{-(c_r l_r^2 + c_f l_f^2)}{\tilde{J}V^2}, \quad (5)$$

$$b_{11} = \frac{c_f}{\tilde{m}V}, b_{12} = \frac{c_f l_f}{\tilde{J}}, \quad (6)$$

$$b_{11} = \frac{c_f}{\tilde{m}V}, b_{12} = \frac{c_f l_f}{\tilde{J}}, \quad (7)$$

FIGURE 1 Lateral vehicle model for lane keeping application



Lane Detection

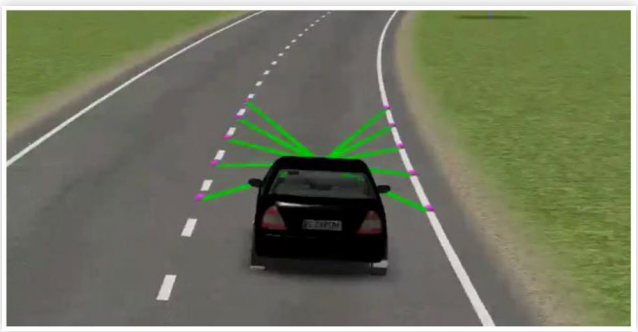
Lane lines on the road can be used to localize the ego vehicle on the road Cartesian coordinates. Our research automated driving vehicle that we plan to use in the future for experimental evaluation has a Mobileye camera which can provide the coefficients of the polynomial fit for the lane detections, lane detection availability and quality information and a GreyPoint camera with our own algorithms for the same outputs. In this paper, we are using our connected and automated driving HIL simulator which has CarSim Real Time with Sensors and Traffic. The CarSim soft camera sensor in the HIL simulator provides the lane detection in the form of x, y coordinates (Figure 2). To simulate the real sensor output and extrapolate the lane detection points, two second order curves denoted by *yl(x)* and *yr(x)* are fitted to the left and right lane detection points respectively, coming from the CarSim software (Equations 8 and 9).

$$y^l(x) = a_0^{\,l} + a_1^{\,l} * x + a_2^{\,l} * x^2 \quad (8)$$

$$y^r(x) = a_0^{\,r} + a_1^{\,r} * x + a_2^{\,r} * x^2 \quad (9)$$

By inserting a longitudinal distance x into the Equations 8 and 9, one can calculate the lateral distance of the vehicle from the right and left lane lines at that longitudinal distance.

FIGURE 2 CarSim soft camera sensor visualization.



Path Generation

For generating the lane level path following map/path, the method presented in [5, 6] is used. This method requires to drive the car at a constant speed at the center of the road and collect accurate GPS data points. These GPS waypoints can also be automatically extracted from a realistic map. We use both approaches. Our future work will be based on an e-Horizon system once we add this capability to our research automated driving vehicle. Collected GPS waypoints are divided into a predetermined number of polynomial segments to capture the different characteristics of the road. These segments are represented as 3rd order parametric polynomials of a distance parameter λ, where λ changes between *i-1* to *i*, according to the number of the segment used. These polynomials are given below as:

$$X_i(\lambda) = a_{xi}\lambda^3 + b_{xi}\lambda^2 + c_{xi}\lambda + d_{xi} \tag{10}$$

$$Y_i(\lambda) = a_{yi}\lambda^3 + b_{yi}\lambda^2 + c_{yi}\lambda + d_{yi} \tag{11}$$

where X_i and Y_i are the path centerline coordinates for the *i*th segment. Since the polynomials fitted to two consecutive segments need to have continuity at their intersection, the polynomials can be fitted to the GPS waypoints using the constrained least squares method. However, to solve the constrained least squares problem, first, the unconstrained problem needs to be solved. The unconstrained problem can be formed in matrix form as shown below.

$$x_{data} = \Lambda n_{x,uncs} \tag{12}$$

$$y_{data} = \Lambda n_{y,uncs} \tag{13}$$

$$\Lambda = \begin{bmatrix} \bar{\lambda}^3 & \bar{\lambda}^2 & \bar{\lambda} & 1 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0 & 0 & 0 & \bar{\lambda}^3 & \bar{\lambda}^2 & \bar{\lambda} & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{14}$$

$$n_{x,uncs} = \begin{bmatrix} a_{x1} & b_{x1} & c_{x1} & d_{x1} & \cdots & a_{xm} & b_{xm} & c_{xm} & d_{xm} \end{bmatrix}^T \tag{15}$$

$$n_{y,uncs} = \begin{bmatrix} a_{y1} & b_{y1} & c_{y1} & d_{y1} & \cdots & a_{ym} & b_{ym} & c_{ym} & d_{ym} \end{bmatrix}^T \tag{16}$$

In the Λ matrix, $\bar{\lambda}$ represents the entire λ vector which ranges from *i-1* to *i*, where *i* is the number of the segment. The number of elements in $\bar{\lambda}$ is equal to the number of the data points in the i^{th} segment. For the given equations, the solution of the unconstrained least square problem is given in Equations 17 and 18 as

$$n_{x,uncs} = \left(\Lambda^T\Lambda\right)^{-1}\Lambda^T x_{data} \tag{17}$$

$$n_{y,uncs} = \left(\Lambda^T\Lambda\right)^{-1}\Lambda^T y_{data} \tag{18}$$

To sustain the continuity and smoothness (continuity of the first derivative) at the segment boundaries, the constraints given below are defined.

$$X_i(i) = X_{i+1}(i) \tag{19}$$

$$Y_i(i) = Y_{i+1}(i) \tag{20}$$

$$\frac{dX_i(i)}{d\lambda} = \frac{dX_{i+1}(i)}{d\lambda} \tag{21}$$

$$\frac{dY_i(i)}{d\lambda} = \frac{dY_{i+1}(i)}{d\lambda} \tag{22}$$

$$\frac{d^2X_i(i)}{d\lambda} = \frac{d^2X_{i+1}(i)}{d\lambda} \tag{23}$$

$$\frac{d^2Y_i(i)}{d\lambda} = \frac{d^2Y_{i+1}(i)}{d\lambda} \tag{24}$$

From the equations 10 and 11, these constraints can be rewritten as shown in Equations 25-30.

$$a_{xi}i^3 + b_{xi}i^2 + c_{xi}i + d_{xi} = a_{xi+1}i^3 + b_{xi+1}i^2 + c_{xi+1}i + d_{xi+1} \tag{25}$$

$$a_{yi}i^3 + b_{yi}i^2 + c_{yi}i + d_{yi} = a_{yi+1}i^3 + b_{yi+1}i^2 + c_{yi+1}i + d_{yi+1} \tag{26}$$

$$3a_{xi}i^2 + 2b_{xi}i + c_{xi} = 3a_{xi+1}i^2 + 2b_{xi+1}i + c_{xi+1} \tag{27}$$

$$3a_{yi}i^2 + 2b_{yi}i + c_{yi} = 3a_{yi+1}i^2 + 2b_{yi+1}i + c_{yi+1} \tag{28}$$

$$6a_{xi}i + 2b_{xi} = 6a_{xi+1}i + 2b_{xi+1} \tag{29}$$

$$6a_{yi}i + 2b_{yi} = 6a_{yi+1}i + 2b_{yi+1} \tag{30}$$

These defined constraint equations are used in matrix form to convert the unconstrained problem into the constrained problem. These equations are combined into a matrix form as shown in Equations 31 and 32.

$$Fn_{x,cs} = 0 \tag{31}$$

$$Fn_{y,cs} = 0 \tag{32}$$

Finally, the solution of the constrained problem is given in Equations 33 and 34.

$$n_{x,cs} = n_{x,uncs} - \left(\wedge^T\wedge\right)^{-1}F^T\left[F\left(\wedge^T\wedge\right)^{-1}F^T\right]^{-1}F_{n,xuncs} \tag{33}$$

$$n_{y,cs} = n_{y,uncs} - \left(\wedge^T\wedge\right)^{-1}F^T\left[F\left(\wedge^T\wedge\right)^{-1}F^T\right]^{-1}F_{n,yuncs} \tag{34}$$

Lateral Deviation Calculation

The lateral controller takes the lateral deviation at a predefined preview distance as input and calculates the corresponding steering angle. As mentioned earlier, two different methods are used to calculate the lateral deviation in this application. The first method uses lane detections acquired from the camera and the second method uses the GPS localization and map based waypoint information.

Lateral Deviation from the Lane Line Detections:

The polynomials representing the lane lines must be parallel to one another as the lane lines are parallel to each other in a real road. Knowing this, the centerline of the road can be represented with the polynomial below in vehicle coordinates.

$$y^c(x) = a_0{}^c + a_1{}^c * x + a_2{}^c * x^2 \tag{35}$$

where coefficients of the polynomial which represents the centerline are given below. Here the superscript "*c*" indicates that the polynomial is a fit for the centerline of the road, and the coefficients of the polynomial are given in Equations 36-38.

$$a_0{}^c = \left(a_0{}^l + a_0{}^r\right)/2 \tag{36}$$

$$a_1{}^c = \left(a_1{}^l + a_1{}^r\right)/2 \tag{37}$$

$$a_2{}^c = \left(a_2{}^l + a_2{}^r\right)/2 \tag{38}$$

Inserting the preview distance l_s into Equation 35 gives the lateral distance of the vehicle at the preview distance.

Lateral Deviation Calculation from the Map and GPS Measurements:

When the lane detections are not available or reliable, the lateral deviation at the preview distance is calculated using the current lateral deviation and the yaw angle error with respect to the generated map. Based on the geometry shown in Figure 1 the lateral deviation at the preview distance l_s can be calculated as

$$y = \mathrm{h} + \mathrm{l}_s \sin(\Delta\Psi) \tag{39}$$

where h is the lateral deviation from the desired path at the vehicle center of gravity, l_s is the preview distance and the $\Delta\psi$ is the yaw angle of the vehicle with respect to the desired path.

First, the lateral deviation of the vehicle from the generated map is calculated. Assuming the radius of the curvature is much larger than the lateral deviation of the vehicle, the shortest distance to the path can be calculated by finding the perpendicular vector to the path from the vehicle center of gravity. This means that the tangent vector of the path will be orthogonal to the shortest vector between the generated path and the vehicle center of gravity as shown in Figure 3. Here the center of the gravity of the vehicle is represented using east and north map coordinates P_E and P_N respectively. Using the fact that dot product of two orthogonal vectors is zero, the solution of Equation 40 for λ_c gives the closest segment position to the vehicle. One can evaluate the x, y coordinates of the closest point on the path and the distance of the vehicle from the path by inserting λ_c into the Equation 41.

$$\left(X(\lambda) - P_E, Y(\lambda) - P_N\right)\left(\dot{X}(\lambda), \dot{Y}(\lambda)\right) = 0 \tag{40}$$

$$h = \rho\sqrt{\left(X(\lambda_c) - P_E\right)^2 + \left(Y(\lambda_c) - P_N\right)^2} \tag{41}$$

where

$$\rho = \mathrm{sgn}\left(\vec{U}(3)\right) \tag{42}$$

$$\vec{U} = \left(\left(X(\lambda_c) - P_E\right), \left(Y(\lambda_c) - P_N\right), 0\right) \mathrm{x} \left(\dot{X}(\lambda), \dot{Y}(\lambda), 0\right) \tag{43}$$

If the third component of the cross product of the path tangent and distance vector is negative, it shows that the vehicle is in the inner side of the desired path and vice-versa.

After finding the h, $\Delta\psi$ is calculated by subtracting the slope of the road at the closest point on the reference path from the yaw angle of the vehicle. Calculation of $\Delta\psi$ can be seen in Equation 44.

$$\Delta\Psi = \Psi - \frac{\dot{Y}(\lambda_c)}{\dot{X}(\lambda_c)} \tag{44}$$

Finally, the lateral deviation at the preview distance can be calculated by inserting h, l_s and $\Delta\psi$ into equation 39.

Lateral Controller Design

The parameter space based design approach given in [7] is used to design the PD controller for the robust lane-keeping controller for the overall system shown in Figure 4. The input and outputs of the system can be listed as the steering command and the lateral deviation at the preview distance respectively. The test vehicle modeled in the state space model of Equation 1 has the numerical parameter values $J = 3{,}728\ \mathrm{kgm}^2$, $C_f = 1.2\mathrm{e}5$ N/rad, $C_r = 1.9\mathrm{e}5$ N/rad, $l_r = 1.5453$ m and $l_f = 1.30$ m where the weight of the vehicle varies between 1,700 kg and 2,000 kg.

FIGURE 3 Position and orientation of the vehicle with respect to the desired path.

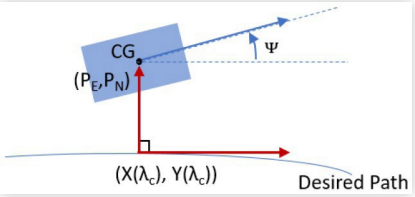
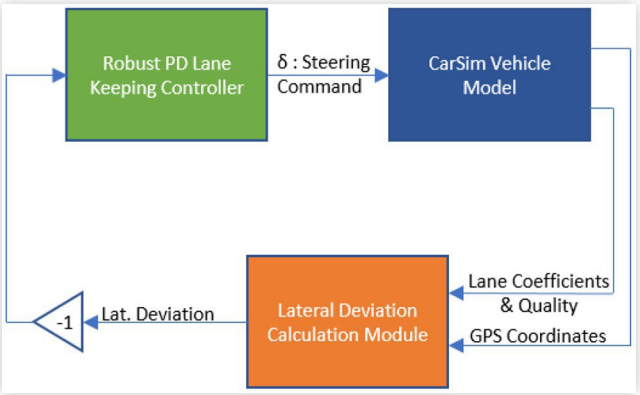


FIGURE 4 Lateral controller system block diagram.



Since the vehicle operates in different load and speed conditions, the controller is designed to be able to work under these different operating conditions as is shown in Figure 5 as an uncertainty box. Also, the preview distance for higher speeds is increased as $l_s = \max(k_s v, l_{smax})$ where v is vehicle speed, k_s is a proportional factor and l_{smax} is the upper bound on the preview length. In this paper, k_s is adjusted such that preview distance changes linearly between 4 m to 7 m for the chosen operating speed range 5 m/s to 30 m/s.

As a D-stability requirement, desired settling time, damping ratio and maximum bandwidth are chosen as 0.5 seconds, 0.7 and 19 rad/sec respectively. PD controller coefficients (K_p and K_d) are chosen as free parameters to find a solution region using the parameter space approach. D-stability solution region is constructed for each corner of the uncertainty box in Figure 5 and they are overlaid on top of each other to find the overall solution region as shown in Figure 6. In this figure blue, green, red, cyan, magenta colored lines show Settling Time Constraint Complex Root Boundary (CRB), Damping Constraint CRB, Bandwidth Constraint CRB, Bandwidth Constraint Real Root Boundary (RRB), Settling Time Constraint RRB respectively. Since blue line is covered by the magenta, it is not clearly visible. Calculation of these boundaries are shown in detail in [7]. By choosing a point in this solution region, one set of K_p and K_d values for the PD controller are chosen as shown in the right plot in Figure 6.

FIGURE 5 Lateral dynamics uncertainty box.

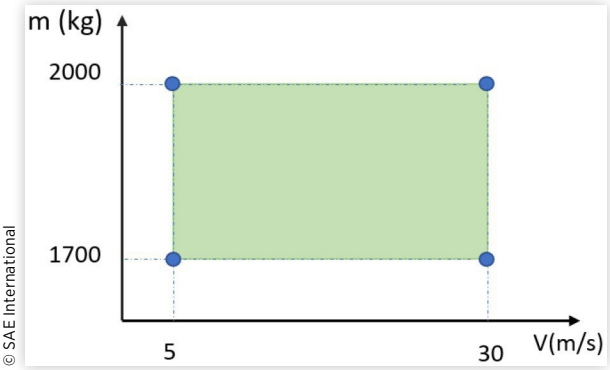


FIGURE 6 Left: Solution regions for the corners of the uncertainty box is plotted on top each other. Right: The zoomed version of the left figure where intersection of the solution regions is highlighted with a gray fill and chosen solution point is shown with a red dot.

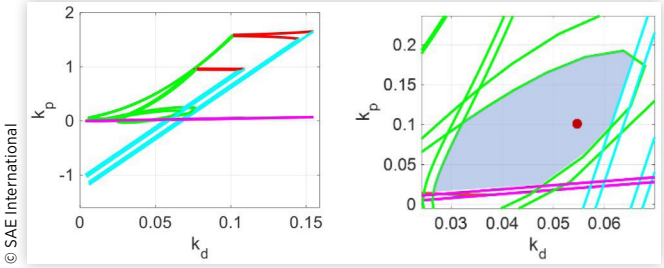
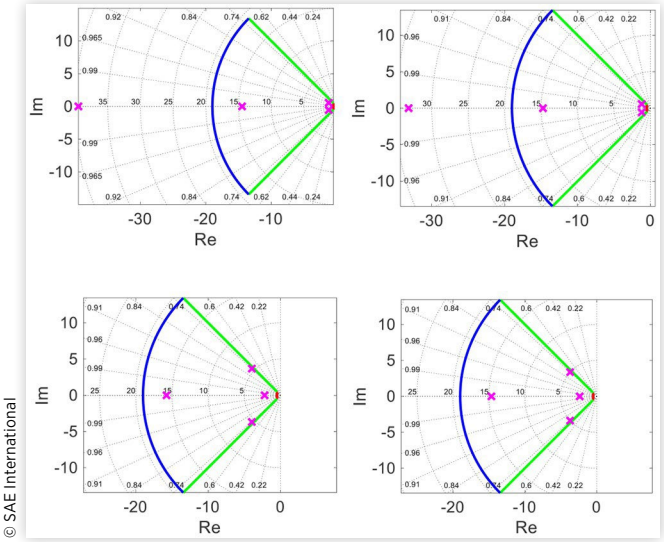


FIGURE 7 D-Stable region and the system pole positions in complex plane for the chosen K_p and K_d. Top Left: 5 m/s, 1700 kg l_s = 4 m, Top Right: 5 m/s, 2000 kg l_s = 4 m, Bottom Left: 30 m/s, 1700 kg /s = 7 m, Bottom Right: 30 m/s, 2000 kg l_s = 7 m.



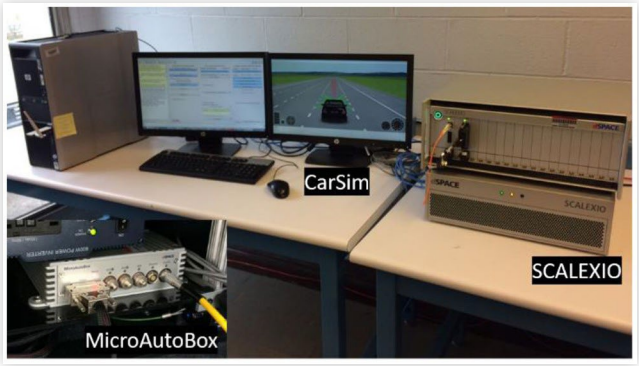
To be considered as a D-stable system, the poles of the system should lie within the D Stable region where it is defined by the desired settling time, the desired minimum damping ratio and the desired maximum bandwidth, all given earlier. As it can be seen from Figure 7, all of the dominant poles of the system which are marked as "x" lie in the D-Stable region for the chosen K_p and K_d coefficients of the PD controller.

Hardware in the Loop Simulator

Testing the developed algorithms in the Hardware in the Loop simulator is a prerequisite to road testing. While testing the vehicles on the road may take extensive time and money, by using simulators, these adverse effects can be minimized with the ease of repeating the simulations in an accident-free environment. Running in real time and being able to connect to the hardware used in the vehicle makes the Hardware in the Loop simulator more advantageous over the regular Model in the Loop simulators. The setup of the HIL simulator used in this paper can be seen in the Figure 8. The setup consists of three main components. The following paragraph will give brief information about these components.

The first component in the system is the computer with CarSim software. This computer is used to design controller algorithms in Matlab and prepare the model of the test vehicle, sensors, and roads. This computer is also used as an interface to communicate with the real-time simulation computer and the controller during the simulation. Secondly, the dSPACE SCALEXIO Processing Unit, which is the real-time computation unit in the HIL, is used to run the validated vehicle model, sensors and traffic information based on the information

FIGURE 8 Hardware in the Loop Simulator setup.



coming from its input ports. Finally, a MicroAutoBox controller unit is connected to HIL, as the control task of the test vehicle is handled by MicroAutoBox. Like the test vehicle setup, in this system computer is connected to MicroAutoBox and the SCALEXIO via ethernet port and the communication between the MicroAutoBox and the SCALEXIO is handled by the CAN Bus. While not used in this paper, we also have DSRC radio units connected as hardware in our HIL simulator. In the next section, simulation results of the lane keeping system are given.

Simulation Results

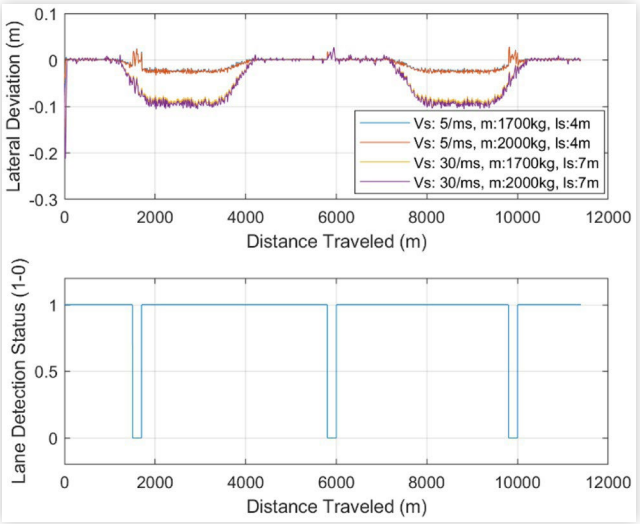
To evaluate the performance of the system a designed lane keeping controller is tested in the CarSim environment described in the previous section. As a test track, a simple model of the high-speed test track in the Transportation Research Center proving ground is constructed in CarSim using its pre-recorded GPS waypoints. The top view of the TRC testing track can be seen in the Google Maps image in Figure 9 which has two curved section between 1000 m-4200 m and 7200 m-10100 m. Since the GPS based path following localization is used as a backup solution, camera based lane keeping system is considered as the active system. In the experimental setup, vehicle is equipped with a Mobileye camera where it can output the quality of the lane line detections. So, in the experiments the mode switching between the camera and GPS will be done based on this lane line detection quality information by the Lateral Calculation Module show in in Figure 4. If the there is no reliable lane detections, vehicle is going to switch to GPS based path following mode. Based on the road conditions lane quality of the system can fail anytime. To simulate the cases where the camera detection fails, the system switches to the GPS based lane keeping mode for pre-defined distance intervals (1500-1700, 5800-6000, and 9800-10000 meters). In the first and third sections vehicle is travelling in the curved parts of the test track while it travels at the straight part of the road in the second interval.

The designed system is tested for the different speed and vehicle mass conditions which are defined as the corners of the uncertainty box given in Figure 5.

The simulation results for the designed system are shown in Figure 10.

FIGURE 9 Top view of the TRC test tracks from Google Maps.

FIGURE 10 Simulation Results for the designed lane keeping system. Top: lateral deviation vs distance traveled for different operating conditions. Bottom: Availability of lane detection vs distance traveled.



When the lateral deviation graph (Figure 10) is analyzed, it can be seen that the designed system still keeps the vehicle in the lane even when the lane detection status goes to zero. Although both the vision based and the GPS based solutions have a higher error for the curved sections of the road, which increases for high speeds, this error is less than 12 cm.

Summary/Conclusions

This paper presented the use of a GPS based lane keeping/path following application as a backup to the camera based lane keeping application. By using these two methods, lane level

control for the vehicle is sustained even when one of the sensor inputs is not available or is not reliable. This happens for instance when lane markings are missing or are very vague or not observable due to weather conditions in certain parts of the road. As it can be seen from the simulation results in Figure 10 both of the designed systems keep the vehicle in the lane accurately. Although increasing the operating speed increases the lateral deviation, especially for the curved parts still the deviation is under 12 cm for the highest vehicle speed of 30 m/s. Also, the system works in different operating conditions where the vehicle weight and the speed are varied over the uncertainty box. As a future work, this simulator will be improved for level 3 to 4 autonomous driving scenario simulations and presented work will be tested in the TRC test track shown in Figure 9.

References

[1]. SAE International, "Automated Driving: Levels of Driving Automation Are Defined in New SAE International Standard J3016," https://www.sae.org/misc/pdfs/automated_driving.pdf, accessed Oct. 2017.

[2]. Ö. Tunçer, L. Güvenç, F. Coşkun and E. Karsligil, "Vision Based Lane Keeping Assistance Control Triggered by a Driver Inattention Monitor, "*2010 IEEE International Conference on Systems, Man and Cybernetics*, Istanbul, 2010, 289-297. doi:10.1109/ICSMC.2010.5642254

[3]. C. M. Kang et al., "Lateral Control for Autonomous Lane Keeping System on Highways," *15th International Conference on Control, Automation and Systems (ICCAS)*, Busan, 2015, 1728-1733. doi:10.1109/ICCAS.2015.7364643

[4]. S. Yenikaya, G. Yenikaya, and E. Duven, "Keeping the vehicle on the road: A survey on on-road lane detection systems," *ACM Comput. Surv.*, 46, 2:1-2:43, July 2013.

[5]. M. Emirler, H. Wang, B. Aksun Güvenç, L. Güvenç. "Automated Robust Path Following Control Based on Calculation of Lateral Deviation and Yaw Angle Error," *ASME. Dynamic Systems and Control Conference*, Volume 3, Columbus, OH, 2015. doi:10.1115/DSCC2015-9856.

[6]. Rossetter, E.J., "A Potential Field Framework for Active Vehicle Lanekeeping Assistance," PhD thesis, Stanford University, 2003.

[7]. L. Guvenc, B. Aksun-Guvenc,. B. Demirel, M.T. Emirler, Control of Mechatronic Systems. IET, 2017.

Contact Information

Automated Driving Lab
930 Kinnear Road, Columbus, OH 43214
cantas.1@osu.edu, guvenc.1@osu.edu

Acknowledgments

This paper is based upon work supported by the National Science Foundation under Grant No.:1640308 for the NIST GCTC Smart City EAGER project UNIFY titled: Unified and Scalable Architecture for Low Speed Automated Shuttle Deployment in a Smart City, by the U.S. Department of Transportation Mobility 21: National University Transportation Center for Improving Mobility (CMU) sub-project titled: Smart Shuttle: Model Based Design and Evaluation of Automated On-Demand Shuttles for Solving the First-Mile and Last-Mile Problem in a Smart City and the Ohio State University Center for Automotive Research Membership Project titled: Use of OSU CAR Connected and Automated Driving Vehicle HiL Simulator for Developing Basic Highway Chauffeur and Smart City Autonomous Shuttle Algorithms.

Authors would like to thank to NVIDIA for their GPU donation to Automated Driving Lab. Simulations presented in this work is done with a computer equipped with a GeForce - GTX TITAN graphics card.

The first author of this paper would like to thank his colleagues in the Automated Driving Lab at the Ohio State University for their support and the Ministry of National Education of the Republic of Turkey for partially supporting his education.

Definitions/Abbreviations

ADAS - Advanced Driver Assistance Systems

GPS - Global Positioning System

RTK - Real-Time Kinematic

HIL - Hardware in the Loop

CRB - Complex Root Boundary

RRB - Real Root Boundary

PD - Proportional, Derivative

CAN - Controller Area Network

DSRC - Dedicated Short Range Communication

TRC - Transportation Research Center

ISSN 0148-7191