Fuzzy Logic Based Self-Driving Racing Car Control System

Berk Korkmaz, Ugur Bugra Etlik, Aykut Beke, Tufan Kumbasar
Department of Control and Automation Engineering
Istanbul Technical University
Istanbul, Turkey
{korkmazbe, etlik, beke, kumbasart}@itu.edu.tr

***Abstract*—In this study, we propose a fuzzy logic based self-driving car control system and its deployment into the JavaScript Racer game. In the design of the self-driving system, it is aimed to design an intelligent vehicle control system such that the racing car finishes the race in the shortest time without getting out of the road. To accomplish such a goal, the proposed intelligent driving system takes up and merges the frameworks of control theory, fuzzy logic and computer vision. In this structure, computer vision is used for lane detection while the fuzzy logic control is employed for positioning and regulating the speed of the racing car. The paper presents in detail the structure and design of the intelligent driving system by presenting all of its design steps. The efficiency of the fuzzy logic based self-driving racing car control system is shown with results collected from the game environment.**

Keywords—fuzzy logic; expert systems, self-driving car; lane tracking; JavaScript Racer

I. INTRODUCTION

In recent years, self-driving car projects have grown in numbers significantly and earned an important place in the automotive industry [1]. However, the self-driving car technology still have some drawbacks such as reliability and cost. In real world applications, the implementation of self-driving car systems is quite challenging and requires very expensive hardware [2]. As a consequence, researchers have handled simulated environments such as computer games to test their self-driving car systems since games provide dynamic, cheap and realistic testbeds. In this context, various games have been investigated such as The Open Racing Car Simulator (TORCS) [3-6], JavaScript Racer [7], VDrift [8], and World Rally Championship 6 [9]. In literature, various types of self-driving car systems have been designed and employed into racing game environments. For example, a nonlinear model predictive control has implemented to VDrift [8] while a fuzzy logic based autonomous vehicle control system has been designed for TORCS [6]. In most of these applications, it is assumed that the race track and lanes are known in advance which separates them from real-word application. Yet, in real-world application, the lanes of the road must be detected in real time. Therefore, vision based lane detection methods are commonly employed [10]. In vision based lane detection, there are two main problems that are lane detection and lane tracking. Detecting a lane is a challenging problem since the road has changing conditions [11]. Many different vision based lane detection systems have been developed and they usually use different lane patterns and different techniques such as hough transform, template-matching, and neural networks [12-14]. On the other hand, lane tracking is also an important problem which is needed for the lateral control of the self-driving car. Various methods have been proposed to generate position reference trajectory based on lane tracking such as fuzzy logic [6], deep reinforcement learning [7], and neural network [14] have been employed to solve this problem.

In this study, we will propose a fuzzy logic based self-driving car control system and its deployment into the JavaScript Racer game. In the design of the self-driving system, we aim to design an intelligent vehicle control system such that the racing car is capable to finish the race in the shortest time without getting out of the road (i.e. as Formula 1 qualifying race). The proposed structure is composed of three main components which are the vision based lane detection system, Fuzzy logic based Position Reference (FPR) generator and Fuzzy logic based Velocity Reference (FVR) generator. We will firstly design the vision based lane detection system to solve road curve estimation problem in real time. The intelligent FPR and FVR structures are designed to generate the position reference (p_r) and velocity reference (v_r), respectively, for the self-driving car control system by processing the road curvature information that is provided by the vision based localization. Then, we will design PD controllers for both the position and velocity control loops of racing car. Finally, in order to show the effectiveness of the developed fuzzy logic based self-driving car control system simulation results are presented and investigated. The game performance of the developed fuzzy logic based self-driving car control system can be seen at https://youtu.be/SO076c2GJg0

This paper is organized as follows. Section II provides information about the JavaScript Racer game. Section III presents the proposed fuzzy logic based self-driving racing car system. The real-time game performance of the proposed intelligent control system are given in Section IV. Finally, the conclusions are given in Section V.

II. JAVASCRIPT RACER GAME ENVIRONMENT

In this study, the JavaScript Racer game environment, which is illustrated in Fig. 1, has been handled [15]. The JavaScript Racer is an open-source pseudo-3d web browser based racing game. It provides a realistic and easily modifiable environment for the self-driving car systems. In the game, various types of roads are available that have different sharpness and length.

978-1-5386-7641-7/18/$31.00 ©2018 IEEE

These adjustable features provide realistic and persuasive testbeds for self-driving systems.

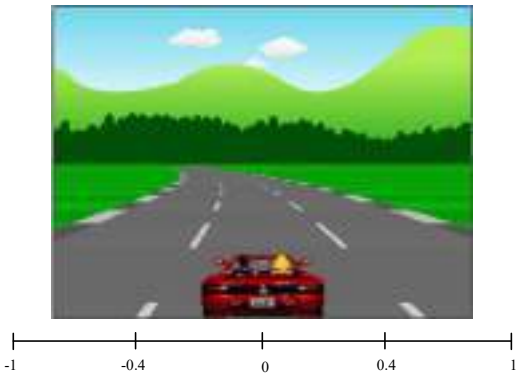


Fig. 1. A screenshot from JavaScript Racer game

In the JavaScript Racer game environment, we cannot directly send velocity or angle reference input signals to the car, we can only send discrete actions. There are four discrete binary actions that are the left (u_L), right (u_R), up (u_U) and binary down (u_D) as given in Table I. u_U and u_D binary key-press actions are used to accelerate and decelerate the car in longitudinal axis, respectively, while u_L and u_R binary key-presses actions are used to accelerate and decelerate the car in lateral axis. Moreover, the velocity of the racing car is limited in between $0 - 12000$ and the visible frame of the image is limited in between -1 (left edge of the road) and 1 (right edge of the road) as seen in Fig. 1. Also, the game has seven road curves that were defined as follows: Left Hard (LH), Left Medium (LM), Left Easy (LE), Straight (S), Right Easy (RE), Right Medium (RM) and Right Hard (RH).

TABLE I. BINARY ACTIONS OF JAVASCRIPT RACER GAME

Input	Action
$u_U \in \{0,1\}$	Longitudinal Up
$u_D \in \{0,1\}$	Longitudinal Down
$u_R \in \{0,1\}$	Lateral Right
$u_L \in \{0,1\}$	Lateral Left

The frames of the JavaScript Racer game environment cannot be collected directly since the JavaScript Racer is a web browser based game. Thus, Tornado web server [16] is used to collect JSON-encoded frames and send JSON-encoded actions to the racing car as seen in Fig. 3. Moreover, we modified the code of JavaScript Racer to send the instant position (p_c) and velocity (v_c) to the our Python web server which would run data through our intelligent control system and return each actions back to the browser in every $T = 0.01s$.

III. FUZZY LOGIC BASED CAR CONTROL SYSTEM

In this section, we will present the fuzzy logic based self-driving racing car control system which is illustrated in Fig. 3.

A. Vision Based Lane Detection

In this subsection, we will explain how we accomplished identifying lane problem with simple but efficient color based image processing methods. The Tornado web server provides us RGB frames with 480×360 pixels resolution. In order to reduce the computational complexity, we have resized the frame into 300×300 pixel frame. Moreover, we have specified a trapezoidal region of interest on this frame that is sufficient for identifying various types of curves present in the game. Then, a perspective transform is performed on this frame.

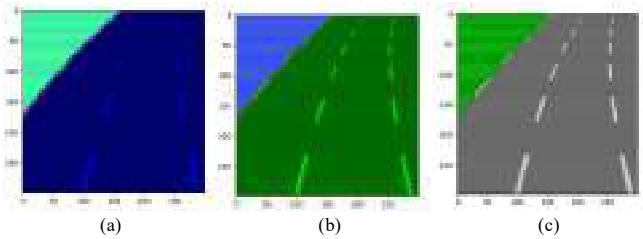


Fig. 2. Transformed images in (a) I_{HSV}, (b) I_{HSL}, (c) I_{RGB} form

In this study, we preferred to use RGB (I_{RGB}), HSV (I_{HSV}) and HSL (I_{HSL}) color spaces as shown in Fig. 2. Then, we applied the following thresholds to increase the performance of lane detection

- HSV mask filter's threshold range is between $(0, 0, 187) - (255, 20, 255)$.
- Mask filter for HSL is defined between $(0, 195, 0) - (255, 255, 60)$
- RGB mask filter is determined between $(200, 200, 200) - (255, 255, 255)$ for white extraction.

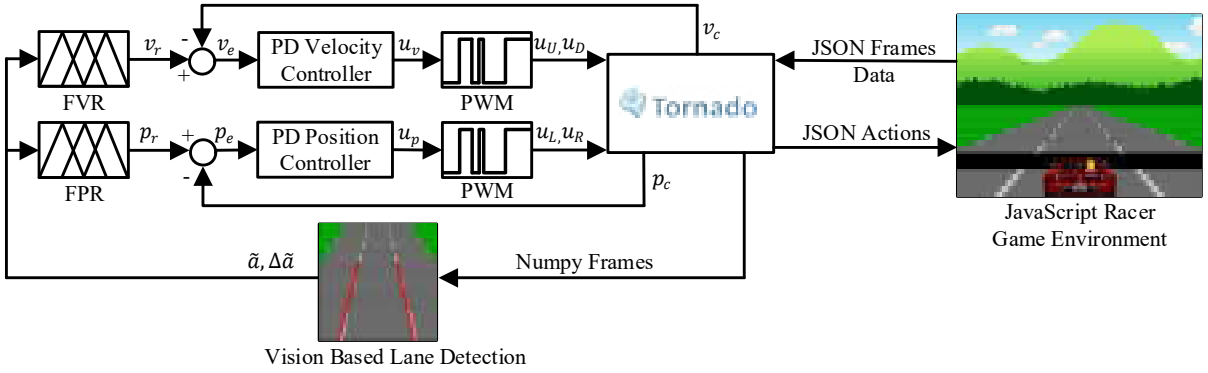


Fig. 3. Fuzzy logic based self-driving racing car control system

The resulting color spaces HSV ($\tilde{I}_{HSV}$), HSL ($\tilde{I}_{HSL}$), and RGB ($\tilde{I}_{RGB}$) are then merged into a final bitmask ($\tilde{I}_F$) as follows:

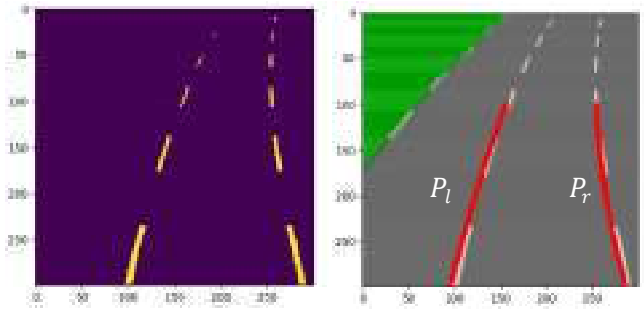
$$\tilde{I}_F = \tilde{I}_{HSV} \vee \tilde{I}_{HSL} \vee \tilde{I}_{RGB} \quad (1)$$

Then, the resulting image given in Fig. 4a is used for lane detection. The lanes were detected by using a sliding histogram window approach on the frame ($\tilde{I}_F$). At each slice, a point was captured where the pixel intensities are the highest. These captured points are then used to fit a 2nd order polynomials that represent the curves. As shown in Fig. 4b, two polynomials are defined one for left line P_l and one for right line P_r that are defined as follows:

$$P_l(x) = a_l\, x^2 + b_l\, x + c_l \quad (2)$$

$$P_r(x) = a_r\, x^2 + b_r\, x + c_r \quad (3)$$

For all possible road curves (LH, LM, LE, S, RE, RM and RH) in the racetrack, we have performed several experiments and calculated corresponding coefficients of P_l and P_r. The mean values of these coefficients are given in Table II.



(a) (b)

Fig. 4. Image $\tilde{I}_F$ (a), visualization of P_l *and* P_r on road (b)

TABLE II. THE COEFFICIENTS OF P_r AND P_l

	LH	*LM*	*LE*	*S*	*RE*	*RM*	*RH*
a_l ($\times 10^{-5}$)	60	35	10	−15	−30	−55	−80
b_l	−0.5	−0.45	−0.35	−0.25	−0.1	0.09	0.21
c_l	200	180	160	140	120	100	80
a_r(10^{-5})	80	55	3	15	−10	35	−60
b_r	−0.23	−0.12	0.09	0.18	0.33	0.39	0.44
c_r	230	207	183	160	137	114	90

As it can be observed from Table II, there is a strong relationship between the coefficients a_l & a_r and the sharpness of curves as expected due to the meaning of 1st and 2nd order derivative tests. Thus, in order to estimate sharpness of the upcoming curve on the frame, we only take account the current mean values of a_l and a_r and its change with respect to time. The mean value (a_m) is defined as

$$a_m = (a_l + a_r)/2 \quad (4)$$

while, the change a_m (Δa_m) is defined as

$$\Delta a_m(k) = a_m(k) - a_m(k-1)/T \quad (5)$$

Here, $T = 0.01s$ is the sampling time. In order to reduce the effect of uncertainties caused by low resolution, we have implemented a moving average filter on the signals a_m and Δa_m.

$$\bar{a}_m(k) = (\sum_{k=1}^{3} a_m(k-3))/3 \quad (6)$$

$$\Delta\bar{a}_m(k) = (\sum_{k=1}^{3} \Delta a_m(k-3))/3 \quad (7)$$

Then, to avoid possible errors while the car is changing its lane in high speed, a low pass filter with $\beta = 0.8$ has been also employed to $\bar{a}_m$ and $\Delta\bar{a}_m$ as follows:

$$\tilde{a}(k+1) = \beta\, \tilde{a}(k) + (1-\beta)\, \bar{a}_m(k) \quad (8)$$

$$\Delta\tilde{a}(k+1) = \beta\, \Delta\tilde{a}(k) + (1-\beta)\, \Delta\bar{a}_m(k) \quad (9)$$

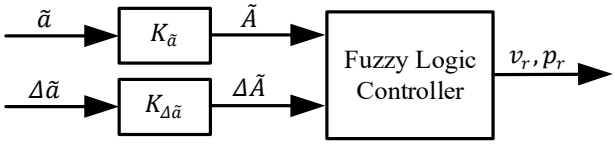


Fig. 5. Fuzzy logic reference generator structure

B. *Intelligent Velocity Control System*

In this subsection, we will present the structure of FVR generator and the conventional velocity controller.

1) Fuzzy Logic Based Reference Velocity Generator

We will present the FVR that generates the proper velocity reference signal (v_r) by using the inputs $\tilde{a}$ and $\Delta\tilde{a}$ as seen in Fig. 5. The inputs are normalized into the universe of discourse of the antecedent Membership Functions (MFs) as follows [17, 18]:

$$\tilde{A} = K_{\tilde{a}}\, \tilde{a} \quad \Delta\tilde{A} = K_{\Delta\tilde{a}}\, \Delta\tilde{a} \quad (10)$$

where, $K_{\tilde{a}}$ and $K_{\Delta\tilde{a}}$ are the input Scaling Factors (SFs) and fixed to the values $K_{\tilde{a}} = 1428.57$ and $K_{\Delta\tilde{a}} = 42.86$.

TABLE III. THE RULE BASE OF THE FVR GENERATOR

$\tilde{a}$ / $\Delta\tilde{a}$	*LH*	*LM*	*LE*	*S*	*RE*	*RM*	*RH*
LH	LV	LV	LV	LV	MV	HV	VHV
LM	LV	LV	LV	MV	HV	VHV	HV
LE	LV	LV	MV	HV	VHV	HV	MV
S	LV	MV	HV	VHV	HV	MV	LV
RE	MV	HV	VHV	HV	MV	LV	LV
RM	HV	VHV	HV	MV	LV	LV	LV
RH	VHV	HV	MV	LV	LV	LV	LV

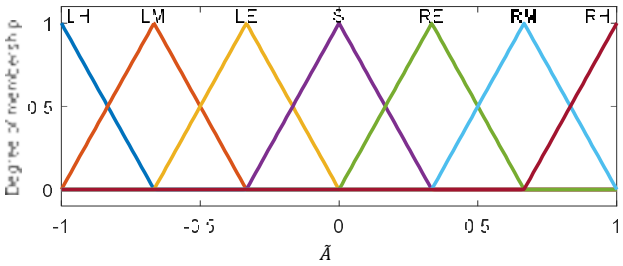


Fig. 6. Illustration of the antecedent MFs of $\tilde{A}$

The FVR generator is constructed with a 7×7 rule base as given in Table III. As shown in Fig. 6 and Fig. 7, the antecedent part of the fuzzy rules is defined with triangular MFs and are defined with same linguistic variables that define the curve types of the game. The consequent part of the fuzzy rules are defined with the following four linguistic fuzzy terms as follows: Low Velocity $(LV) = 4000$, Medium Velocity $(MV) = 5000$, High Velocity $(HV) = 6000$, and Very High Velocity $(VHV) = 7000$. The FVR uses and employs the product implication and the center of sets defuzzification method [17, 18]. The resulting surface of FVR can be seen in Fig. 8.

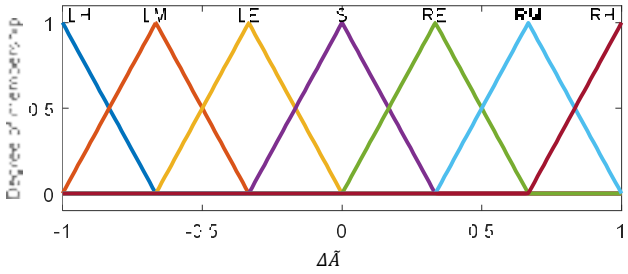


Fig. 7. Illustration of the antecedent MFs of $\Delta\tilde{A}$

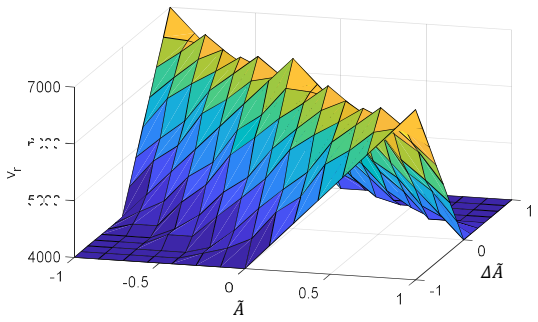


Fig. 8. Illustration of the FVR generator surface

2) PD Velocity Controller

We have firstly performed a system identification experiment to reveal the relationship between the car velocity (v_c) and the control signal (u_v). The system dynamics are defined with the following first order integrating difference equation:

$$v_c(k) = v_c(k-1) + 65.97\, u_v(k-1) \tag{11}$$

To regulate the velocity of the racing car, we have designed a PD controller to minimize the velocity error (v_e) defined as

$$v_e(k) = v_r(k) - v_c(k) \tag{12}$$

The control law of PD in sampled time domain is

$$u_v(k) = K_p\, v_e(k) + K_d\, (v_e(k) - v_e(k-1))/T \tag{13}$$

where $T = 0.01$s and the coefficients of PD controller are tuned as $K_p = 0.00001656$ and $K_d = 0.0000001$ to have a fast system response. As it can be seen in Fig. 3, the sampled control signal u_v is then converted into the discrete game actions $u_U \in \{0,1\}$ and $u_D \in \{0,1\}$ via a Pulse Width Modulation (PWM) generator as follows:

$$\begin{aligned} u_v > 0 &\rightarrow u_U = 1, u_D = 0 \\ u_v < 0 &\rightarrow u_U = 0, u_D = 1 \end{aligned} \tag{14}$$

The resulting closed loop system performance for velocity reference $v_r = 7000$ is given in Fig. 9.

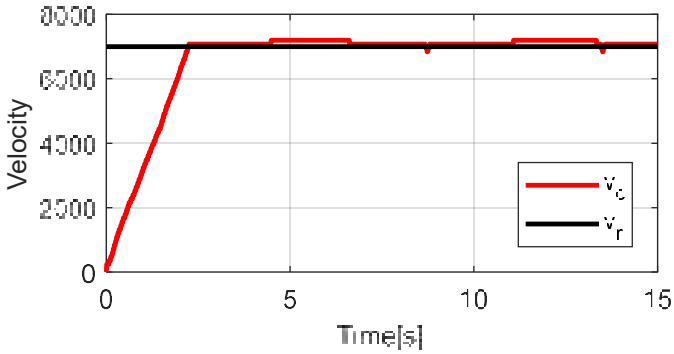


Fig. 9. PD controller performance

C. Intelligent Position Control System

In this subsection, the design of the FPR generator and position control system for the system will be explained.

1) Fuzzy Logic Based Reference Position Generator

The FPR generator uses $\tilde{a}$ and $\Delta\tilde{a}$, similar to the FVR structure, to generate position reference p_r for the racing car as shown in Fig. 5. The FPR generator is constructed with a 7×7 rule base as given in Table IV. The FPR uses the same antecedent MFs of the FVR, while the consequent part is defined with five linguistic fuzzy terms as follows: Far Left Position $(FLP) = -0.7$, Close Left Position $(CLP) = -0.35$, Middle Position $(MP) = 0$, Close Right Position $(CRP) = 0.7$, and Far Right Position $(FRP) = 0.7$. The FVR uses and employs the product implication and the center of sets defuzzification method [23]. The resulting surface of FPR generator is given in Fig. 10.

TABLE IV. THE RULE BASE OF THE FPR GENERATOR

$\tilde{a}$ / $\Delta\tilde{a}$	*LH*	*LM*	*LE*	*S*	*RE*	*RM*	*RH*
LH	FLP	FLP	FLP	FLP	MS	CLP	MP
LM	FLP	FLP	FLP	MS	CLP	MP	CRP
LE	FLP	FLP	FLP	CLP	MP	CRP	FRP
S	FLP	FLP	CLP	MP	CRP	FRP	FRP
RE	FLP	CLP	MP	CRP	FRP	FRP	FRP
RM	CLP	MP	CRP	FRP	FRP	FRP	FRP
RH	MP	CRP	FRP	FRP	FRP	FRP	FRP

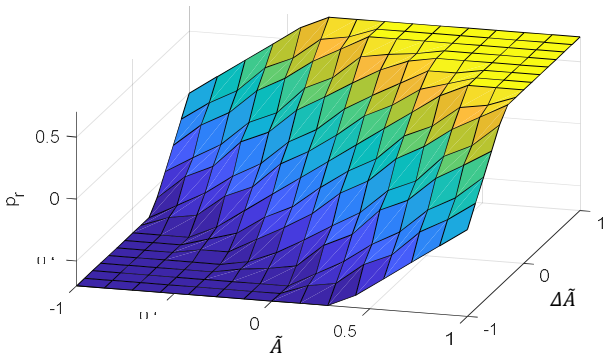


Fig. 10. Illustration of the FPR generator surface

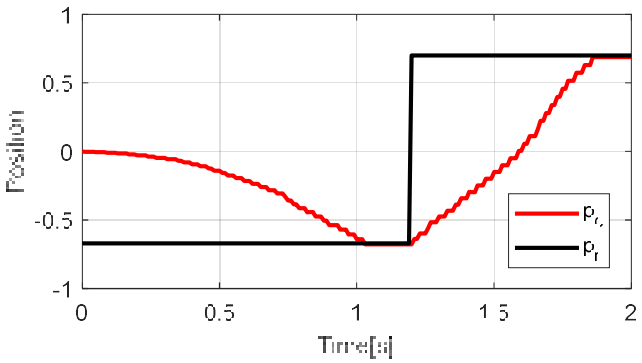


Fig. 11. PD controller behavior

2) PD Position Controller

We have firstly performed a system identification experiment to obtain the difference equation between the car position (p_c) and the control signal (u_p).

$$p_c(k) = 1.98\,p_c(k-1) - 0.98\,p_c(k-2) + 10^{-5}(8.82\,u_p(k-1) + 8.76\,u_p(k-2)) \quad (15)$$

To regulate the position of the racing car, we have designed a PD controller to minimize the position error (p_e) defined as:

$$p_e(k) = p_r(k) - p_c(k) \quad (16)$$

The sampled control law of the PD controller is as follows:

$$u_p(k) = K_p\,p_e(k) + K_d\,(p_e(k) - p_e(k-1))/T \quad (17)$$

The coefficients of PD controller are tuned as $K_p = 5$ and $K_d = 0.05$. As we have done in the velocity control system, the control signal u_p is then converted into the binary game actions $u_R \in \{0,1\}$ and $u_L \in \{0,1\}$ via a PWM generator as follows:

$$\begin{aligned} u_p > 0 &\rightarrow u_L = 0, u_R = 1 \\ u_p < 0 &\rightarrow u_L = 1, u_R = 0 \end{aligned} \quad (18)$$

The resulting position control performance for varying position reference signal is given in Fig. 11.

IV. Real-Time Game Performance

Here, we will examine the real-time game performance of the proposed fuzzy logic based self-driving racing car control system. In the experiment, the car is going to complete a racetrack that is structured with all possible road curves (LH, LM, LE, S, RE, RM and RH) which are generated in a random order. The resulting intelligent control system performance is given in Fig. 12. It can be firstly concluded that the overall racing performance of the proposed autonomous control system is satisfactory. It can be also stated that when the racing car is positioned on one of the lanes, the vision based lane detection mechanism is unable to estimate the curves with polynomials efficiently. This effect can be seen at 8.8s and 23.5s in the Fig. 12. However, these effects are eliminated in a short time. The game performance of the intelligent control system can be seen at https://youtu.be/SO076c2GJg0

Now, let us focus on the shaded area in Fig. 12a, the car is driving through a LH curve at 26.5ths to a RH curve at 29ths. Consequently, due to the radical change of road curve characteristics, the FPR mechanism changed the p_r value from -0.7 to 0.7. The car tracked this position reference change within 2.6s. Moreover, as it can be seen in the shaded area in Fig. 12b, during the transition from LH curve to RH curve, the v_r value has increased until 28.5ths to stabilize the position dynamics of the car. After that, the v_r value has decreased to an optimum value for avoiding effects of centrifugal force.

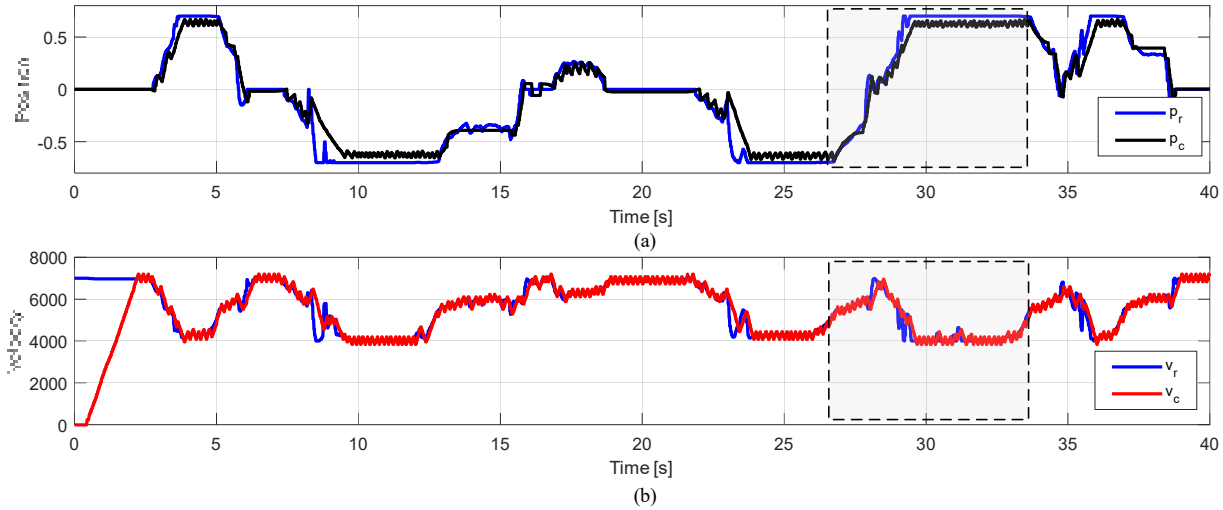


Fig. 12. Fuzzy logic based self-driving racing car control system performance

V. CONCLUSION AND FUTURE WORK

In this study, we proposed a self-driving racing car control system and its performance was examined in JavaScript Racer game environment. Firstly, we designed vision based lane detection system to solve lane identification problem. The predicted lane curvature information was used for lane tracking. For this problem, we proposed intelligent velocity and position control systems. FPR and FVR generators were designed by using expert knowledge to generate reference values. Experimental results show that the proposed fuzzy logic based self-driving racing car control system has reached the desired reference values very quickly and they eliminated effects of disturbances in a short period of time..

For our future work, we plan to deploy the proposed intelligent structure to more complex computer games and to a real-world autonomous car.

REFERENCES

[1] T. Litman, "Autonomous vehicle implementation predictions," Victoria Transport Policy Institute, 2017.

[2] R. W. Wall, J. Bennett, and G. Eis, "Creating a low-cost autonomous vehicle," *IEEE 2002 28th Annual Conference of the Industrial Electronics Society (IECON02)*, vol. 4, pp. 3112-3116, 2002.

[3] T. Wang and K. T. Liaw, "Driving style imitation in simulated car racing using style evaluators and multi-objective evolution of a fuzzy logic controller," 2014 IEEE Conference on Norbert Wiener in the 21st Century (21CW), Boston, pp. 1-7, 2014.

[4] S. Zuo, Z. Wang, X. Zhu, and Y. Ou, "Continuous reinforcement learning from human demonstrations with integrated experience replay for autonomous driving," *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Macau, pp. 2450-2455, 2017.

[5] D. Perez, G. Recio, and Y. Saez, "Evolving a fuzzy controller for a car racing competition," IEEE Symposium on In Computational Intelligence and Games, pp. 263-270, 2009.

[6] E. Armagan and T. Kumbasar, "A fuzzy logic based autonomous vehicle control system design in the TORCS environment," *10th International Conference on IEEE*, pp. 737-741, 2017.

[7] A. Yu, R. Palefsky-Smith, and R. Bedi, "Deep reinforcement learning for simulated autonomous vehicle control," Course Project Reports: Winter, 1-7, 2016.

[8] F. Kehrle, J. V. Frasch, C. Kirches, and S. Sager, "Optimal control of formula 1 race cars in a VDrift based virtual environment," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11907-11912, 2011.

[9] E. Perot, M. Jaritz, M. Toromanoff, and R. d. Charette, "End-to-end driving in a realistic racing game with deep reinforcement learning," *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Honolulu, pp. 474-475, 2017.

[10] S. G. Jeong et al., "Real-time lane detection for autonomous vehicle," *IEEE International Symposium on Industrial Electronics Proceedings, Pusan,* vol. 3, pp. 1466-1471, 2001.

[11] A. A. Assidiq, O. O. Khalifa, M. R. Islam and S. Khan, "Real time lane detection for autonomous vehicles," *International Conference on Computer and Communication Engineerin*, pp. 82-88, 2008.

[12] Y. Wang, E. K. Teoh, and D. Shen, "Lane detection and tracking using B-Snake," *Image and Vision Computing*, vol. 22, no. 4, pp. 269-280, 2004.

[13] F. You, R. Zhang, G. Lie, H. Wang, H. Wen, and J. Xu, "Trajectory planning and tracking control for autonomous lane change maneuver based on the cooperative vehicle infrastructure system," *Expert Systems with Applications*, vol. 42, no. 14, pp. 5932-5946, 2015.

[14] Z. Kim, "Robust lane detection and tracking in challenging scenarios," *in IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 16-26, 2008.

[15] J. Gordon (2016, January 6). Javascript pseudo 3D racer [Online]. Available: http://github.com/jakesgordon/javascript-racer.

[16] "Documentation", Computer Vision System Toolbox Documentation. [Online]. Available: http://www.tornadoweb.org/en/stable/. [Accessed: 15-January-2018].

[17] A. I. Savran, A. Beke, T. Kumbasar, E. Yesil, "An IMC based fuzzy self-tuning mechanism for fuzzy PID controllers," *In Innovations in Intelligent SysTems and Applications (INISTA)*, pp. 1-7, 2015.

[18] F. Candan, A. Beke, T. Kumbasar "Design and Deployment of Fuzzy PID Controllers to the nano quadcopter Crazyflie 2.0," in *2018 IEEE INISTA*, 2018.