A Lateral Control Based on Physics Informed Neural Networks for Autonomous Vehicles

Tamás Hegedűs[1(✉)], Dániel Fényes[1], Balázs Németh[1], Vu Van Tan[2], and Péter Gáspár[1]

[1] HUN-REN Institute for Computer Science and Control (SZTAKI), Kende u. 13-17, Budapest 1111, Hungary
{tamas.hegedus,daniel.fenyes,balazs.nemeth, peter.gaspar}@sztaki.hun-ren.hu

[2] Department of Automotive Mechanical Engineering, Faculty of Mechanical Engineering, University of Transport and Communications, Cau Giay Street, Hanoi 100000, Vietnam
vvtan@utc.edu.vn

Abstract. In the paper, a lateral control strategy is presented using Physics-Informed Neural Network (PINN) for automated vehicles. The main idea is that the physics information is incorporated into the training process, which leads to an improvement in the performance level of the control algorithm. Moreover, in the highly nonlinear range of the lateral dynamics, which is not properly covered by the training dataset, the stability of the vehicle is guaranteed. The results are compared to a conventional neural network trained to control the vehicle.

Keywords: Automated vehicles · Neural networks · Lateral control

1 Introduction

In general, the modeling process of an arbitrarily chosen system often relies on large datasets, which must cover the whole operational range of the system. This can be a challenging task, and the presence of nonlinearities and uncertainties makes it even harder. However, the modeling phase is essential since the reachable performance level highly depends on the accuracy of the model in several applications such as control and observer design.

In the field of control design, the applied methods can be sorted into two main groups, based on how the collected dataset is utilized. The first group contains the classical methods, in which the dataset is used for constructing a nominal mathematical model. Whilst the second group consists of methods, which directly use the data points during the control design [1]. In practice, the machine learning-based solutions cannot be used for control purposes in safety-critical systems. However, combined control structures give an option for the control-oriented use of machine learning-based methods [5].

G. Mastinu et al. (Eds.): AVEC 2024, LNME, pp. 812–818, 2024.
https://doi.org/10.1007/978-3-031-70392-8_115

Although the combined solutions can guarantee the robustness of the control loop, the neural network-based layer must have a high accuracy, which can achieved using non-conventional training methods such as Physics-Informed Neural Network (PINN). The main idea behind this approach is to consider the physical information of the system during the training process [2,7]. The goal of the paper is to present a high-performance level neural network for combined control structures. In the paper, a comparison study can be found between the conventional and the modified training process of the network. The whole method is validated through an automated vehicles-motivated problem: trajectory tracking.

The paper is structured as: In Sect. 2 the data generation process, and the nonlinear system are presented. In Sect. 3 the physics-informed neural network is detailed in terms of control purposes. The results of the simulations can be found in Sect. 4. Finally, the paper is summarized in Sect. 5.

2 System Description and Data Generation

In this section, the mathematical formulation of the vehicle dynamics and the data acquisition of the training process is presented. The main equations of the vehicle dynamics rely on the lateral forces, which can be computed from the slip angles (α_i), where $i = \{f, r\}$, f denotes the front, while r is the rear axis [8]. The axle positions measured from the center of gravity are denoted by l_i. Using these values and the tire characteristics (Magic formula), the lateral forces of both the front and the rear tires can be computed using [6]:

$$F_{y,i} = mgl_{f,r}L^{-1}\mu_i sin(c \cdot tan^{-1}(b \cdot \alpha_i)). \tag{1}$$

The shape of the nonlinear tire characteristics is determined by the parameter c, b. The distance between the rear and front axis is: $L = l_f + l_r$. The vehicle mass is m, while the gravitational acceleration is g. Based on the tire characteristics and the slip angles, the lateral forces can be computed for the axles. The vehicle motion is described by two main equations:

$$\frac{d^2\psi}{dt^2} = \frac{1}{I_z}(l_f F_{y,f} - l_r F_{y,r}), \qquad \frac{dv_y}{dt} = \frac{1}{m}(F_{y,f} + F_{y,r}), \tag{2}$$

where $\dot{\psi}$ is the angular velocity (yaw-rate), and the lateral velocity is given by v_y. The goal is to calculate the derivatives during the training process of these signals, which can be done using the continuous dynamical equations: $\dot{x} = \mathcal{F}(x_0, u, p_i)$, where $\mathcal{F}$ is the dynamical system, x_0 gives the initial conditions, u is the control input and the varying parameters are given by p_i. In practice, the continuous systems are discretized to fulfill the requirements of the implementation [3], for the time-varying system: $x(k+1) = A(k)x(k) + B(k)u(k), y(k) = C^T(k)x(k)$. Using the discrete state space model of the system, the output can be predicted based on the input signals sequence ($\mathcal{U} = [u(1), u(2)...u(N_p)]$). The tracking error can be computed from the predicted output and the reference value ($y_{ref} \in \mathbb{R}^{n \times N_p}$) as $\epsilon(k) = y_{ref}(k) - y(k)$. Then, using the predicted error signal, the following quadratic optimization can be formed [3]:

$$\min_{\mathcal{U}} \sum_{i=1}^{N_p} \left(\gamma(y_{ref}(i) - y(i))^2 + \lambda u^2(i)\right), \text{ s.t.} \quad \phi\mathcal{U} < b \quad \text{and} \quad l_b \leq u \leq l_u, \tag{3}$$

where $\gamma \in \mathbb{R}$ and $\lambda \in \mathbb{R}$ are weights, which make the balance between the tracking accuracy and the energy used for the control purposes. The upper and lower bounds for the control input signal are denoted by l_b, l_u. Moreover, a bound is defined for the chosen, predicted states of the system (b), and ϕ can be constructed from the system matrices.

In the next step, the data generation process is presented. Considering that the tire characteristics significantly depend on the operational point of the vehicle, the nonlinear system is linearized at each operational point. Furthermore, it is assumed, that the reference longitudinal velocity is known over the prediction horizon. The data generation process is carried out by driving the vehicle along the reference trajectory, which is defined as:

$$y_{ref}(t) = \beta(p_1 x(t)^3 + p_2 x(t)^2 + p_3 x(t) + p_0) sin(\omega t), \tag{4}$$

where t provides the simulation time. The parameters of the reference lateral position are varied randomly to generate trajectories with different radius. The parameters of the reference positions are selected to: $p_1 \in [-1, 1], p_2 \in [-2, 2], p_3 \in [-0.5, 0.5], \omega \in [0, 0.5]$, $\beta = 0.1$. The sampling time is set to $T_s = 0.02\,\text{s}$, and the longitudinal velocity is set between 10 and 20 m/s.

Firstly, the initial states and the parameters of the reference trajectory are randomized. Secondly, the vehicle is driven along the route solving the optimization problem (3), which calculates the control input signal sequence, and several signals are saved during the simulation. The measured states of the vehicle are: $x = [y_0,\ v_y,\ \dot{\psi},\ \psi,]^T$, and the control input is the steering angle (δ). Moreover, the output of the system is the lateral position. During the data generation the weights are set to $\gamma = 1, \lambda = 5$ to ensure smooth trajectory tracking. Moreover, the yaw-rate ($\dot{\psi}$) is limited to $0.7\,\text{rad/s} \geq |\ \dot{\psi}\ |$ and the limits for the control input is set to $0.2\,\text{rad} \geq |\ u\ | \ \forall u \in \mathcal{U}$ to ensure stable motion requirements. Finally, the prediction horizon is selected to $N_p = 45$.

3 Physics Informed Neural Network in Control Structure

In Fig. 1, the whole training structure is presented of the neural network. The input vector consists of the initial states of the vehicle (x_0). Since the neural network is implemented for trajectory tracking, the input vector is augmented with a sequence of the reference signal ($y_{ref,1} ... y_{ref,k}$), where k denotes the length of the reference signal. Moreover, taking into account

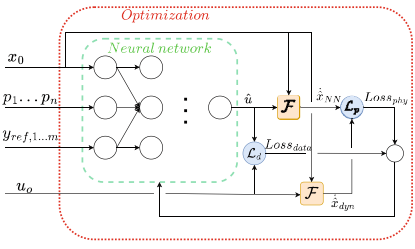


Fig. 1. Structure of the training process

the varying parameters of the system $(p_1 ... p_n)$, these values are also incorporated into the input vector. In practice, the horizon length for the reference signal and for the changing parameters should be chosen to the same. Finally, the u_{MPC} gives the reference control signal value, which is computed from (3). In Figure $\mathcal{L}$ gives the computation method of the loss function within one iteration step.

3.1 Computation of the Loss Function

The loss function, which is the base of the optimization process, is built up from two main parts: the data-based part and the physics-based part. Firstly, the data-based loss function is detailed. In every iteration step, the predicted output signal of the neural network is denoted as $\hat{u}$ and computed as:

$$\hat{u} = \hat{F}(x_0, p_{1,2...n}, y_{ref,1}, y_{ref,2}, y_{ref,m}), \tag{5}$$

where $\hat{F}$ is the approximated function of the system. Using the predicted value of the network, for the given set of training datasets the error between the target and the predicted data can be computed as $\epsilon_{data} = \hat{u} - u_{MPC}$. Secondly, the physics-based loss function is calculated. The derivative(s) of the specific states can be computed using the nonlinear function of the system (2). Using the initial states, the varying parameters, and the predicted neural network output: $\dot{\hat{x}} = \mathcal{F}(x_0, \hat{u}, p_i)$. Since the real output of the neural network is known (u_{MPC}), the real derivatives can be also computed by the nonlinear description of the systems. The calculated error between the derivatives is crucial during the training process. This part of the algorithm takes into account the nonlinear effects of the dynamics for the error between the real and the results of the neural network. This makes the estimation more accurate within the ranges, where a small deviation between the real and the estimated value results in a significant error in the changes of the system states. This effect is observable, especially within the ranges influenced by highly nonlinear effects. This makes the performance level of the neural network higher within these ranges, which makes the neural network more reliable. Both the data-based loss function $(Loss_{data})$, and the physics-based part $(Loss_{phy})$ is computed using L_2 loss:

$$\mathcal{L}_{NN} = Q_1 \frac{1}{N} \sum_{i}^{N} ||(\epsilon_{data,i})||^2 + \frac{1}{M} \sum_{j}^{M} Q_j \frac{1}{N} \sum_{i}^{N} ||\dot{\hat{x}}_{i,j} - \dot{x}_{i,j}||^2, \tag{6}$$

where M gives the number of the computed derivatives and Q_1, Q_i aims to scale the loss values to each other. These values are computed for each mini-batch of the neural network and also computed for every epoch. The training process of the neural network can be made using an optimizer such as the ADAM optimization algorithm [4]. In the following, the effectiveness of the proposed training process is demonstrated through a lateral trajectory tracking problem of automated vehicles.

3.2 Training Dataset and the Neural Network

The whole training dataset consists of 60000 data points, which are saved during the 4000 randomly generated test scenarios. The output of the neural network is computed through the input vector (5). In the vehicle-oriented implementation, the input vector is $T = [v_y, \psi, \dot{\psi}, v_{1,p}, \epsilon_{1...n}]$. ϵ_i gives the predicted error between the current lateral position and the reference trajectory. However, for training purposes, not the whole reference trajectory is used, but it is sampled along the prediction horizon as $i = \{1, 12, 23, 34, 45\}$. Moreover, the first and the predicted last longitudinal velocity is used. Since the goal is the trajectory tracking, the output of the network is the steering angle. During the training process, based on the nonlinear differential equations, the predicted derivatives of the lateral displacement and the yaw-rate are considered (see: (2)) in the data-based part of the loss function. Moreover, the weights in the loss function are $Q_{data} = 10, Q_{\ddot{\psi}} = 5, Q_{\dot{v}_y} = 1$.

The size of the input vector of the neural network is 10, while the output is 1. The network has 3 hidden layers and the number of neurons is 25, 40, 20. Moreover, the activation functions are Hyperbolic tangent, ReLU, and Hyperbolic tangent. The training process of the network is carried out using the Adaptive Moment Estimation (ADAM) algorithm [4]. During the optimization the Learning rate is selected to 0.01, the size of the mini batch is 256, while the maximum epoch number is 200.

4 Simulation Results

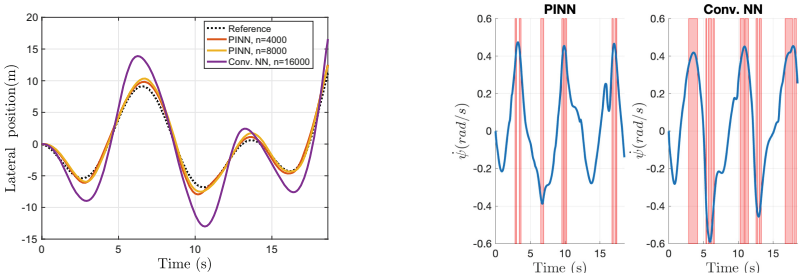
In this section, the conventionally trained and the PINN are compared to each other in terms of tracking performance. The test scenarios are implemented in CarMaker vehicle dynamics simulation software, in which a Tesla Model S vehicle is used. For training purposes, a small amount of data is used to highlight the advantage of the incorporation of physical information. Moreover, dynamical ranges are eliminated from the training dataset in terms of yaw-rate: $(0.35 \leq |\dot{\psi}| \leq 0.45) \vee (0.55 \leq |\dot{\psi}| \leq 0.65)$. The results are summarized in Table 1.

Table 1. Test results using different neural networks

Size of data	$n = 2000$	$n = 4000$	$n = 8000$	$n = 12000$	$n = 16000$
PINN	✗	✓	✓	✓	✓
Conv. NN	✗	✗	✗	✓	✓

The test is said to be successful if the vehicle fulfills the stable motion requirements during the scenario. In Table 1, successful tests are marked with a checkmark (✓), while unsuccessful tests are marked with a crossmark (✗). Considering the results shown in the table, it can be observed, that the PINN-based solution was successful in reference trajectory tracking with the use of 4000 data points,

whereas the conventionally trained network required at least 12000 data points. Furthermore, the lateral positions are depicted in Fig. 2a, for three cases: PINN, with 4000 and 8000 training data points, and the conventionally trained with 12000 points. Although the conventional trained model used more data points, the physics-based extension in the loss function increased tracking accuracy as shown in Fig. 2a. In Fig. 2b the yaw-rates during the trajectory tracking can be examined. The regions, which are not covered by the training dataset, are highlighted in red.



(a) Lateral positions during test scenarios (b) Yaw-rates of the vehicle

Fig. 2. The velocity and the trajectory of the vehicle

5 Conclusion

The paper presented a novel PINN-based lateral control strategy for autonomous vehicles. The physics-based information of the vehicle has been utilized during the training process of the vehicle. As the simulation examples have shown the this information can significantly improve the performance level of the controller and guarantee the stable motion even under extreme circumstances.

Acknowledgement. The research was supported by the National Research, Development and Innovation Office through the project "Cooperative emergency trajectory design for connected autonomous vehicles" (NKFIH: 2019-2.1.12-TÉT_VN). The paper was partially funded by the National Research, Development and Innovation Office under OTKA Grant Agreement No. K135512. The work of Daniel Fenyes was supported by the Janos Bolyai Research Scholarship of the Hungarian Academy of Sciences.

References

1. Aradi, S.: Survey of deep reinforcement learning for motion planning of autonomous vehicles. IEEE Trans. Intell. Transp. Syst. **23**(2), 740–759 (2020)
2. Arnold, F., King, R.: State-space modeling for control based on physics-informed neural networks. Eng. Appl. Artif. Intell. **101**, 104195 (2021)

3. Grune, L., Pannek, J.: Nonlinear Model Predictive Control. Springer, London (2011)
4. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2017)
5. Nemeth, B., Hegedus, T., Gaspar, P.: Performance guarantees on machine-learning-based overtaking strategies for autonomous vehicles. In: 2020 European Control Conference (ECC), pp. 136–141 (2020)
6. Pacejka, H.: Tire and Vehicle Dynamics. Elsevier Science (2005)
7. Raissi, M., Perdikaris, P., Karniadakis, G.: Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. **378**, 686–707 (2019)
8. Rajamani, R.: Vehicle Dynamics and Control. Springer, New York (2005)