IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 19, 2021, accepted September 7, 2021, date of publication September 14, 2021,
date of current version September 22, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3112560

Model Predictive Control With Learned Vehicle Dynamics for Autonomous Vehicle Path Tracking

MOHAMMAD ROKONUZZAMAN, NAVID MOHAJER, SAEID NAHAVANDI, (Fellow, IEEE), AND SHADY MOHAMED

Institute for Intelligent Systems Research and Innovation (IISRI), Deakin University, Waurn Ponds, VIC 3216, Australia

Corresponding author: Mohammad Rokonuzzaman (mpappu@deakin.edu.au)

ABSTRACT Model Predictive Controller (MPC) is a capable technique for designing Path Tracking Controller (PTC) of Autonomous Vehicles (AVs). The performance of MPC can be significantly enhanced by adopting a high-fidelity and accurate vehicle model. This model should be capable of capturing the full dynamics of the vehicle, including nonlinearities and uncertainties, without imposing a high computational cost for MPC. A data-driven approach realised by learning vehicle dynamics using vehicle operation data can offer a promising solution by providing a suitable trade-off between accurate state predictions and the computational cost for MPC. This work proposes a framework for designing an MPC with a Neural Network (NN)-based learned dynamic model of the vehicle using the plethora of data available from modern vehicle systems. The objective is to integrate an NN-based model with higher accuracy than the conventional vehicle models for the required prediction horizon into MPC for improved tracking performances. The proposed NN-based model is highly capable of approximating latent system states, which are difficult to estimate, and provides more accurate predictions in the presence of parametric uncertainties. The results in various road conditions show that the proposed approach outperforms the MPCs with conventional vehicle models.

INDEX TERMS Autonomous vehicles, path tracking controller, model predictive control.

I. INTRODUCTION

Path Tracking Controller (PTC) is an integral subsystem of an Autonomous Vehicle (AV). For designing PTC for AVs, the complexity and fidelity of the chosen vehicle model vary depending on the type of the tracking task and control technique [1]. For example, to implement the path tracking at low speed, a simpler vehicle model is sufficient and provides reasonable accuracy [2]. A kinematic vehicle model is a popular choice for designing PTC for the lower-speed operation of AVs [3]–[5]. However, due to unmodelled dynamics, a kinematic vehicle model is not viable at higher speeds [2]. In these situations, a dynamic vehicle model is more accurate. Different fidelity vehicle dynamic models such as bicycle and dual-track models have been used depending on the complexity of path tracking tasks.

A number of techniques have been proposed to design the PTC [1], [6]. Geometry-based controllers such as Pure Pursuit [7] and Stanley controller [8] are quite popular due to their low computational cost, ease of implementation and acceptable performance at low speeds. Direct Lyapunov-based controller [9] and feedback linearisation-based controller [10] have been also found practical solutions for PTCs of AV. These controllers generally do not consider the dynamics effects of the vehicle and suffer from a high level of uncertainties, so they are not suitable for the AVs in the urban environment [2]. To reduce the effect of the unmodelled vehicle dynamics, dynamic vehicle models are used to design PTCs. These types of controllers include Sliding Mode Control (SMC) [11], Optimal Control [12], Adaptive Control [13] and Model Predictive Control (MPC) [14]–[16].

MPC has shown to be a suitable option for AVs that can accommodate controller design with a reasonable level of complexity and computational cost. In addition, this control approach has the potential to ensure and increase the comfort of the AVs' passengers by improving handling performance of an AV [17]. One of the major advantages of MPC is its ability to handle multiple variables and constraints. Besides, it has inherent robustness against uncertainties. It can address the physical limit of the actuators making it highly useful for AV's PTC design. Moreover, additional constraints on the

The associate editor coordinating the review of this manuscript and approving it for publication was Qing Yang.

states can be imposed to ensure the safety and stability of the vehicle. Nevertheless, the performance of the controller directly depends on the accuracy of the vehicle model and requires careful considerations. As MPC predicts the states of the vehicle for a certain time horizon at each sampling time, the accuracy of these predictions affects the performance of the controller significantly. A high fidelity dynamic vehicle model can improve the performance of the controller. However, the computational cost associated with a complex vehicle model may not be suitable for real-time operation as MPC solves an optimisation problem at each time step.

For PTCs, two different approaches are generally adopted to design MPCs. These include Linear MPC (LMPC) using a linearised vehicle model [14], [18], [19] and Nonlinear MPC (NMPC) where a fully nonlinear model is utilised [15], [20]. For linearisation of vehicle dynamics, successive linearisation at each operating point is a common approach that transforms the model into a linear time-varying model [14], [18], [19]. However, the use of linearised vehicle model is only applicable for certain operating regions. For example, the force approximation using linear tire model becomes invalid for large slip angles [14], [21].

In conventional MPC design, uncertainties are either tackled by designing robust controllers or by estimating the values of the parameters. A robust controller can handle uncertainties up to a certain limit where a bound on the uncertainty needs to be known. In this regard, robust MPC such as tube-based MPC has been proposed by researchers. In the tube-based MPC approach [22], a feedback controller is used to keep the state within an invariant tube even under the influence of uncertainties. This approach has been used for the active safety of the vehicle by ensuring the state and input constraints are satisfied in the presence of disturbances and uncertainties due to model mismatch [23]–[26].

The nonlinear dynamic model provides more accuracy; however, it still may not completely capture the dynamics of a vehicle. The design of an analytical mathematical model generally requires choosing some specific physical aspects that are most significant for the control task and ignore others. However, the efficacy of these choices depends on the designer's capability and the required control task. A simpler model may perform well for some specific situations, yet, in some cases, the unmodelled dynamics may introduce uncertainty and significantly affect the controller's performance. On the other hand, a highly complex model may not be the best option to be used in the MPC context due to the computation cost of the online optimisation. In this regard, a learning-based MPC where the vehicle dynamics are learned using the vehicle operation data can provide a suitable trade-off between accuracy, unmodelled dynamics and complexity.

Different types of vehicles are currently available, and a general model formulation for all vehicle types is difficult. Even for the same kind of vehicles, some properties will be inherently different. Besides, several subsystems in a vehicle affect the motion, such as the steering, brake and suspension systems, which may also need to be integrated into the vehicle model in some capacity. For instance, involving the steering dynamics has been recommended to improve the performance of MPC [27], [28]. Furthermore, some aspects change due to the operating conditions, including parametric and non-parametric uncertainties affecting vehicle motion. For example, different environmental conditions such as the friction coefficient of the road surface, wind speed, vehicle weight and load transfer also impact vehicle motion. In addition, some vehicle parameters change during the vehicle lifetime. A rigidly defined vehicle model may not be suitable for different vehicles in various operation conditions. As the design and development of AV are becoming more advanced and optimised and eventually adopted by more people, an adaptive data-driven approach may be required to identify and design vehicle dynamic model.

A Neural Network (NN) is a highly capable solution for approximating nonlinear functions and can be used for learning a vehicle dynamic model using the measured state and input data of the vehicle. A properly designed and trained NN does not generally suffer from unmodelled dynamics and provides more accurate performances. Besides, it can handle the parametric uncertainties given that it is trained with sufficient data. For any dynamic system, identifying of latent system states is demanding and generally circumvented by estimating some parameters. However, identification of these states is not necessary when an NN-based approach is adopted. A properly trained NN with sufficient data can identify its internal representation of time-varying dynamics [29]. In addition, a NN trained with state and input history can identify variation in latent states such as vehicle load and friction co-efficient.

The NN-based model identification has been used for controller design in different systems. This approach has been used for controlling the helicopters [30]–[32], autopilot control of aerial vehicles [33], underwater vehicles [34], [35], and different industrial systems such as wastewater treatment [36], interface level in a flotation column [37] and PH maintaining system [38]. In the context of AV, NN-based system identification has been adopted in a number of reported researches. For example, in [39], [40], this approach was used for identifying longitudinal dynamics, and in [41], it was used for modelling the steering dynamics. A more detailed combined lateral and longitudinal dynamics vehicle model was designed using this approach in [42]. For the control of AVs, a number of control techniques have been adopted with an NN-based system model. For instance, a backstepping variable mode controller was reported in [43] and a sliding mode fuzzy controller was proposed in [44]. Furthermore, more recently, feedforward control [45] and iterative LQR [46] with NN-based vehicle model is proposed.

In the context of AV, the NN-based vehicle model identification approach has been proposed in combination with other control techniques. However, the use of this model in the MPC for PTC design has been unexplored. MPC can be potentially used to improve the path tracking performance

using a more accurate NN-based vehicle model. Especially with the modern vehicle data acquisition system providing abundant operational data, a learning-based approach can provide a more reliable solution for vehicle dynamics approximation. In this work, we propose a new data-driven MPC where two sets of states and input measurements history are maintained for a few previous steps for NN prediction. The first set contains the controlled vehicle's state and input measurements, and the second set is used for the predicted states and corresponding inputs during the MPC optimisation. These histories with current measurements are used to estimate future states. This approach allows more accurate prediction in the presence of uncertainties in the vehicle's parameters, such as surface friction coefficient and load variation. The use of state and input measurements history allows more accurate predictions up to a specific prediction horizon.

The main contributions of this work include: (1) Learning a more accurate vehicle dynamic prediction model than the current analytical vehicle models using a NN. The learned model can be reliably used in MPC to provide more accurate state estimations up to a certain prediction horizon without significantly increasing the MPC computational cost. (2) Demonstrating that the resulted MPC with an NN-based prediction model can improve the tracking accuracy of an AV in the presence of parametric uncertainty. (3) Designing a novel Switched MPC (SMPC) with an adaptive NN-model where the NN's weights and biases are updated online using vehicle measurements data. In the switching scheme, a choice between a nonlinear analytical model and the adaptive NN model is made based on a cost function. We propose two different approaches for designing MPC with an NN-based lateral vehicle model. In the first approach, the NN is trained offline with the data collected for various operating conditions and used by the MPC for the prediction of the states. In the second approach, an adaptive technique is adopted for training NN where the network's weights and biases are updated based on real-time data from the vehicle. An SMPC is used to accommodate the use of the online trained NN-model. Ultimately, the performances of the proposed approaches are compared to the existing LMPC and NMPC.

The rest of the paper is organised as follows. In section II, a discussion of the AV and the conventional physical models used for MPC design is provided. In section III, the details of the NN-based vehicle model are discussed. Next, the design of MPC using the NN-based vehicle model is reported in section IV. The implementation procedure is addressed in section V and the performances of the proposed vehicle models and the controllers are evaluated on various conditions, and the results are reported in section VI. Finally, the conclusion of the work is drawn in section VII.

II. PRELIMINARIES

This section provides a brief introduction to the AV system, including the chosen states and inputs. A preliminary discussion on the most commonly used vehicle dynamic model and their variation in the MPC context is also provided. These vehicle models are used for comparing the performance of the NN-based vehicle model and the proposed MPC controller.

The considered AV system can be expressed as

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, u_t, \mathbf{w}_\theta), \tag{1}$$

where, t is time, $\mathbf{x}_t \in \mathbb{R}^n$ is the state vector, $u_t \in \mathbb{R}^m$ is the input of the system, and $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the vehicle transition function. In addition, $\mathbf{w}_\theta$ represents a set of variables that represents the parametric uncertainty of the system.

In this work, we limit the study only to parametric uncertainties $\mathbf{w}_\theta$ of the system. For the sake of simplicity, we assume that other forms of uncertainties (including noise) and delays are negligible for the current system. In addition, the state values are assumed to be directly measurable, and state estimations are not required. Our primary objective is to design an MPC with an NN-based vehicle model that provides improved performance under different parametric uncertainties. We assume that certain vehicle parameters such as vehicle load and road surface friction vary during vehicle operation. These parameters differ from their initial values, and the variations in parameters are not known. Here, the objective is to use the NN which can identify the underlying changes in parameter values and provide more accurate predictions for the MPC.

It is aimed to design a lateral MPC controller for which vehicle states $\mathbf{x} = [X, Y, \psi, v_x, v_y, a_y, r]$ are considered. Here, $[X, Y]$ is the vehicle position on the global coordinate, ψ is the yaw angle, v_x is the longitudinal velocity, v_y is the lateral velocity, a_y is the lateral acceleration and r is the yaw rate of the vehicle. The control action is the steering angle $u = \delta$ for the system.

A. DYNAMIC VEHICLE MODEL

The bicycle model is the most commonly used for designing MPC. Here, the considered vehicle has a mass m and a moment of inertia I_z at the vehicle center of gravity. The dynamic model of the vehicle can expressed as [47], [48]

$$\dot{v}_x = \frac{1}{m}\big[F_{xf}\ \cos(\delta) - F_{xb} + F_{yf}\ \sin(\delta)\big] + v_y r, \tag{2a}$$

$$\dot{v}_y = \frac{1}{m}\big[F_{yf}\cos(\delta) + F_{yb} - F_{xf}\sin(\delta)\big] - v_x r, \tag{2b}$$

$$\dot{r} = \frac{1}{I_z}\big[F_{yb} l_b + \big(F_{xf}\sin(\delta) - F_{yf}\cos(\delta)\big) l_f\big]. \tag{2c}$$

Here, the forward and rear wheels are represented by f and b. Besides, F_x and F_y is the longitudinal and lateral force, respectively. In addition, l_f is the distance of the front wheel from the centre of gravity, and l_b is the distance between the rear wheel and the centre of gravity. Finally, δ represents the steering angle of the vehicle. Figure 1 shows the schematics of a vehicle dynamic model.

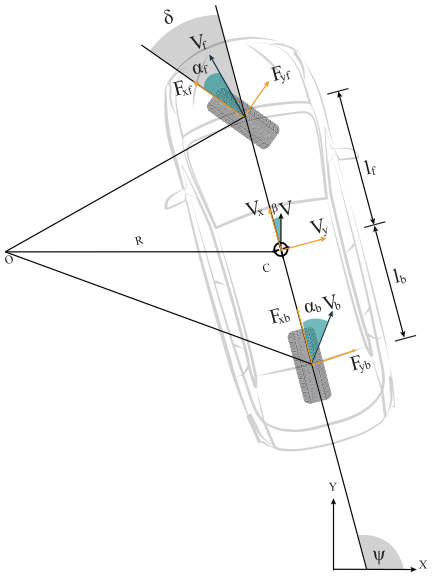


FIGURE 1. Geometry of a dynamic bicycle model of a car-like vehicle.

1) LINEAR TYRE MODEL

Different tyre models have been proposed for designing MPC depending on the slip angle of the vehicle. For small slip angles, a linear tyre model is often used where the relationship between the cornering stiffness and the generated force is linear. For the linear tyre model, the wheel slip angle for the front α_f and rear wheel α_r can be expressed as [47]

$$\alpha_f = \delta - \tan^{-1}\left(\frac{v_y + l_f\dot{\psi}}{v_x}\right), \tag{3a}$$

$$\alpha_b = -\tan^{-1}\left(\frac{v_y - l_b\dot{\psi}}{v_x}\right). \tag{3b}$$

For the small slip angle approximation, a linear relationship between lateral tyre force and tyre slip angle can represented as [48],

$$F_{yf} = -C_f\alpha_f, \tag{4}$$

$$F_{yb} = -C_b\alpha_b, \tag{5}$$

where C_f is the cornering stiffness of the front wheels and C_b is the cornering stiffness of the rear wheel.

2) NONLINEAR TYRE MODEL

The linear tyre model is only efficient for small slip conditions. For larger slip conditions, nonlinear tyre models perform significantly better than linear models. An analytical model such as Brush model [49] is one of the commonly used approaches for approximating vehicle tyre forces. In this approach, the tyre forces are calculated using the wheel's lateral slip angle (α) and the normal force (F_z). In the context of MPC, a modified brush model is often used [50], [51]. For this model, the lateral tyre force is expressed as [49]

$$F_y = \begin{cases} -C_\alpha \tan\alpha + \dfrac{C_\alpha^2}{3\mu F_z}|\tan\alpha|\tan\alpha \\ \qquad -\dfrac{C_\alpha^3}{27\mu^2 F_z^2}\tan^3\alpha & \text{if } |\alpha| < \alpha_s \\ \mu F_z\,\mathrm{sgn}(\alpha) & \text{Otherwise} \end{cases} \tag{6}$$

Here, μ is the tyre-road friction coefficient, C_α is the cornering stiffness, F_z is the normal load and α_s is the slip saturation angle. This α_s is calculated as

$$\alpha_s = \tan^{-1}\left(\frac{3\mu F_z}{C_\alpha}\right) \tag{7}$$

These vehicle models will be used for the performance comparison with the proposed approach. We implement two different MPCs using these vehicle models and compare their performances with MPC with the proposed NN-based vehicle model.

III. LEARNING NEURAL NETWORK VEHICLE MODEL

To learn a lateral vehicle model using a NN, a subset of vehicle states is assumed as $\chi = [v_x, v_y, r]$ and a vector $\mathbf{q}_t = [(\chi_t, \ldots \chi_{t-N_h+1}), (u_t \ldots u_{t-N_h+1})]$ representing the current and history of the states and control value up to a certain time period N_h is considered at each time instance t.

One of the primary objectives of this work is to train a multilayer feedforward NN that provides the following relationship between the current and history of inputs and states to the next step of the system.

$$\hat{\mathbf{y}}_t = f_{NN}(\mathbf{q}_t, \boldsymbol{\omega}) \tag{8a}$$

$$u_t = u_{t-1} + \Delta u_t. \tag{8b}$$

where $\hat{\mathbf{y}}_t = [\dot{v}_{y,t}, \dot{r}_t]$ is the estimated value of the lateral and yaw accelerations of the vehicle. f_{NN} represents the NN with two hidden layers, each with N units and $\boldsymbol{\omega}$ is the sets of weights and biases. In this work, we chose a feedforward multilayer NN even though a number of other NN architecture can be used for this proposed work. Multilayer NN is one of the commonly used architecture for system identification due to the simplicity of its design and ease of implementation. This NN architecture has successfully been used for identification of different system such as helicopter systems [31], aerial vehicle [33] and underwater vehicles [34], [35]. In this work, different hyper-parameters of the NN are chosen based on our previous experience of NN-based designs and trial-and-error. The architecture of the NN is shown in Fig. 2.

Two different approaches are used for training the NN. In the first approach, data from human driving is collected for different road conditions and then used to train the NN offline. In the second approach, the NN is trained online in parallel with the vehicle operation.

A. OFFLINE TRAINED MODEL

The training dataset of n number of trajectories with a sampling step size Δt is assumed as

$$\mathcal{D}^{(i)} = \{\mathbf{q}_t^i, \mathbf{q}_{t-1}^i \ldots . \mathbf{q}_{t-p}^i\} \tag{9}$$

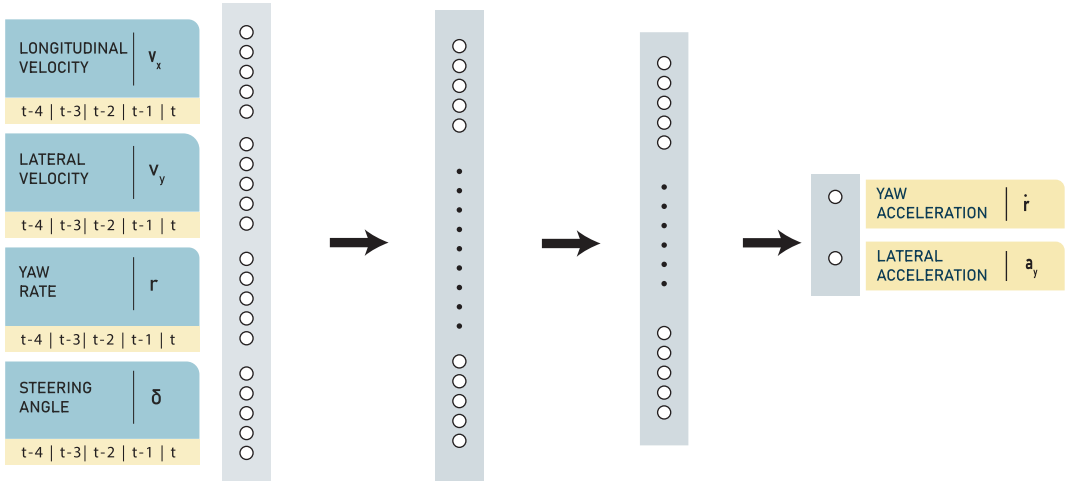


FIGURE 2. The proposed architecture for NN.

Here, $i = 1, 2 \ldots n$ is the trajectory instances and p is the time step in each trajectory.

In the standard NN approach, the objective is to find a set of weights and biases that reduces the error between the estimated network output and the observed data from the system. The NN shown in Fig. 2 establishes the following relationship

$$\hat{\mathbf{y}}_t = \mathbf{W}_{L_2}\phi\big(\mathbf{W}_{L_1}\phi(\mathbf{W}_I\mathbf{a} + \mathbf{b}_1) + \mathbf{b}_2\big) + \mathbf{b}_3 \tag{10a}$$

where, $\mathbf{a} := (\chi, u) \in \mathbb{R}^{|a|}$ is the input the NN and $\phi(\cdot)$ is the activation function. $\mathbf{W_I}$, $\mathbf{W_{L_1}}$, $\mathbf{W_{L_2}}$ represent the weights of the input layer, first hidden layer, and the second hidden layer, respectively. Similarly, $\mathbf{b_1}$, $\mathbf{b_2}$, $\mathbf{b_3}$ represent the weights of the input layer, first hidden layer, and the second hidden layer, respectively. The choice of training algorithms and other hyper-parameters such as the number of neurons in each layer, activation function, and learning rate are discussed in more detail in section V.

The output of the NN can be used to estimate the lateral velocity and yaw rate of the vehicle as follows

$$\begin{bmatrix} v_{y,t} \\ r_t \end{bmatrix} = \begin{bmatrix} v_{y,t-1} \\ r_{t-1} \end{bmatrix} + \hat{\mathbf{y}}_t dt \tag{11a}$$

The trained network is used to predict the states' output in the context of MPC for the lateral control of a vehicle. The formulation of MPC with NN-based vehicle state prediction model will be shown in section IV.

B. ONLINE TRAINED MODEL

An efficient PTC should be able to perform under different operating conditions. Approximating vehicle transition dynamics using the offline trained NN needs a large dataset containing information from different road conditions with different vehicle states and controls. Moreover, a static NN model may not be sufficient for a highly dynamic system such as AV operating on various environmental conditions. To circumvent this, an adaptive approach to learn NN-based model is proposed. Here, the network is trained online when the data from the vehicle is updated.

It is assumed that no data is available prior to the start of the vehicle operation. In this approach, training the NN before the start of the vehicle operation is not required. The NN weights and biases are initialised using the Nguyen-Widrow method (NW) [52] and then updated sequentially when a new set of data is available from the vehicle. The network weights and biases are always updated using the N_s number of the vehicle measurement data. The training of the NN starts when N_s steps of vehicle state measurement and corresponding input data are available from the vehicle operation. The weights and biases are updated periodically after a specific update delay of $d = N_n$ steps, which allows the use of new N_n number of data for each update. A mixture of old and new data $N_s = N_c + N_n$ is used to update the network, where N_c is the number of old data and N_n is the number of new operation data. The change in weight after each iteration can be expressed as

$$\Delta W(t+1) = M\Delta W(t) + (1 - M) * L * W(t) \tag{12}$$

where, ΔW is the change in weights, M is the momentum constant, and L is the learning rate. This operation is conducted in parallel with the path tracking task. More details on the choice of the dataset size and hyperparameter values will be reported in section V.

IV. CONTROLLER DESIGN WITH LEARNED MODEL

The formulation of the MPC for the lateral control of an AV based on the NN-based lateral transition model is discussed in this section. Firstly, the MPC with the offline trained NN-based vehicle model is discussed. Then, a switched MPC approach for the online trained NN-based vehicle is reported.

A. MPC WITH OFFLINE TRAINED MODEL

For the AV system of (1), based on the discussion on section III, the NN-based transition model can be

expressed as

$$\begin{bmatrix}\dot{v}_y & \dot{r}\end{bmatrix}^{\mathsf{T}} = f_{NN}(\mathbf{x}_k, \ldots \mathbf{x}_{k-N_h+1}, u_k \ldots u_{k-N_h+1}) \quad (13a)$$

$$\dot{X} = v_x \cos\psi - v_y \sin\psi \quad (13b)$$

$$\dot{Y} = v_x \sin\psi + v_y \cos\psi \quad (13c)$$

$$\dot{\psi} = r \quad (13d)$$

For the sake of compactness, the vehicle transition model is expressed as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \dot{\mathbf{x}}_k \Delta t = f_{NN}(\mathbf{x}_k, u_k, \boldsymbol{\theta}) \quad (14)$$

where, $\boldsymbol{\theta} = [\mathbf{W}_I, \mathbf{W}_{L_1}, \mathbf{W}_{L_2}, \mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3]$ are the parameters of the NN. In an MPC approach, the optimal control problem is solved using a receding horizon approach. At any time t, the MPC problem can be expressed as

$$\arg\min_{U} \sum_{k=0}^{N_p-1} J(\mathbf{x}_{k|t}, \mathbf{u}_{k|t}) \quad (15a)$$

$$\text{subjected to, } \hat{\mathbf{x}}_{t+k+1|t} = f_{NN}(\hat{\mathbf{x}}_{t+k|t}, u_{t+k|t}, \boldsymbol{\theta}) \quad (15b)$$

$$u_k = u_{k-1} + \Delta u_k, \quad (15c)$$

$$\mathbf{x}(0|t) = \mathbf{x}(t) \quad (15d)$$

$$u(k) \in \mathcal{U} \quad \forall k \in [t, t+N_p] \quad (15e)$$

$$\hat{\mathbf{x}}(k) \in \mathcal{X} \quad \forall k \in [t, t+N_p] \quad (15f)$$

Here, N_p is the prediction horizon, and J is the stage cost. The state and input constraints are represented by sets $\mathcal{X}$ and $\mathcal{U}$. In addition, $\hat{\mathbf{x}}$ represents the predicted state of the vehicle based on the current measured state. At each time step, an optimal solution for the control action $U* = [u_t^*, u_{t+1}^* \ldots .u_{t+N_p}^*]$ is found and only the first control action is sent to the real system. Then, the whole process is repeated at the next time step. Fig. 3 shows the architecture of MPC with the offline trained NN transition model.

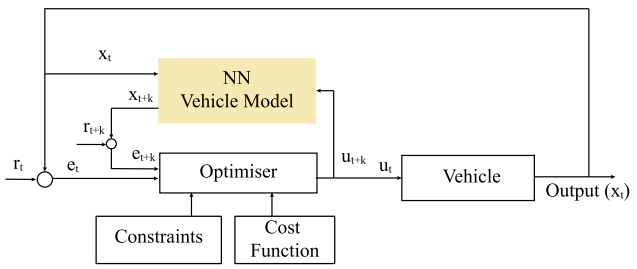


FIGURE 3. MPC with NN-based vehicle model.

During the MPC control process, the future states are predicted for a certain time horizon at each sampling time based on the current state. The NN vehicle transition model requires a history of states and corresponding control actions for certain previous time steps along with the current states and control input. To facilitate this, two separate sets of states and input histories are maintained: 1) history of real vehicle $\mathbf{H}_r$, including vehicle state measurements during vehicle operation, and 2) history of the predictions $\mathbf{H}_p$, calculated using vehicle model during the optimisation process. At each sampling time, the optimisation process is started by replacing $\mathbf{H}_p$ with real vehicle history $\mathbf{H}_r$. These $\mathbf{H}_p$ values are then used for the prediction of the states using the NN model. During the optimisation process, the history of the predicted states $\mathbf{H}_p$ is updated based on the estimated states for the corresponding input generated by the optimiser. The algorithm for the MPC with an NN-based vehicle model is shown in Algorithm 1.

Algorithm 1 MPC With NN-Based Vehicle Model

Input: Initial state $\mathbf{x}_0$ (feedback from real vehicle), history of real vehicle states and input $\mathbf{H}_r = [\mathbf{x}_{t-1}, \mathbf{x}_{t-2} \ldots \mathbf{x}_{t-N}, u_{t-1}, u_{t-2} \ldots u_{t-N}]$, history of estimated state and corresponding input $\mathbf{H}_p$, prediction horizon N_p, cost function J, NN-based vehicle model f_{NN}

```
Form the optimisation problem using (13),(14) and (15)
while MPC is running do
    Measure current state x_t.
    Update H_p = H_r with current state measurement and control (if available).
    Start the optimisation problem
    while Optimisation is running do
        for i = 1:N_p do
            Estimate next state x̂_{t+i} using f_NN, H_p and u_i
            Update H_p using the estimated states and control
        end for
        Find optimal control sequence
    end while
    Apply only u(t|t)
    t = t+1
end while
```

B. SWITCHED MPC FOR ONLINE TRAINED MODEL

In the online training approach, the NN-model is capable of adopting new data collected during vehicle operation. However, this requires a certain number of data (vehicle operation for a certain time) and a certain iteration of learning to be effective. This is true at the start of vehicle operation or when a completely different operating condition is faced. During this time, the network needs to be trained in a number of iterations with new data to perform better than a nonlinear dynamic model.

To circumvent the aforementioned problem, a switched MPC is suggested. In this approach, both the NN-based vehicle model and the nonlinear dynamic model (discussed in section II-A) can be in effect. The vehicle model used for state prediction is switched based on the accuracy of the predictions of these models. At each time interval, prediction from both models is compared with the vehicle's current state using the same input sent to the vehicle. This prediction error is calculated based on the difference between the models' predicted states and the vehicle's current state. In this case,

the switched MPC formulation is expressed as

$$\arg\min_{U} \sum_{k=0}^{N_p-1} J(\mathbf{x}_{k|t}, \mathbf{u}_{k|t}) \tag{16a}$$

$$\text{subjected to } \hat{\mathbf{x}}_{t+k+1|t} = f_p(\hat{\mathbf{x}}_{t+k|t}, u_{t+k|t}) \tag{16b}$$

$$f^p = h(e_p) \tag{16c}$$

$$u_k = u_{k-1} + \Delta u_k \tag{16d}$$

$$\mathbf{x}(0|t) = \mathbf{x}(t) \tag{16e}$$

$$u(k) \in \mathcal{U} \quad \forall k \in [t, t+N_p] \tag{16f}$$

$$\hat{\mathbf{x}}(k) \in \mathcal{X} \quad \forall k \in [t, t+N_p] \tag{16g}$$

Here, f^p is vehicle transition function where p represents either the NN-based model or the nonlinear dynamic model, e_p is the prediction error of each p^{th} vehicle model and $h(\cdot)$ is the switching function. The prediction error is calculated as

$$e_p(f^p, t) = |\hat{\mathbf{x}}_t^p - \mathbf{x}_t|^2 \tag{17}$$

where, $\hat{\mathbf{x}}^p$ is the predicted states using the vehicle model f^p and $\mathbf{x}$ is the measured vehicle state. Fig. 4 shows the architecture of the SMPC with the adaptive NN transition model. The same approach discussed in Algorithm 1 is used if the SMPC selects the NN model.

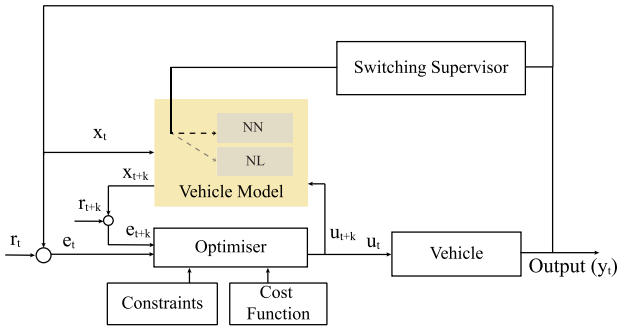


FIGURE 4. Switched MPC with NN-based vehicle model. Here, NN represents the adaptively learned NN-based vehicle model and NL represents the conventional nonlinear analytical vehicle model.

V. IMPLEMENTATION

A simulated testbed using Matlab and the physics simulator 'Unreal Engine' is developed to evaluate the performance of the proposed NN-based MPC controllers. A complex, high fidelity 14 Degree of Freedom (DoF) vehicle model including several other subsystems such as steering, suspension, transmission and driveline is used to simulate the vehicle. Details of the vehicle system are discussed in the following subsection. This model represents an actual vehicle that is too complex to be used in the MPC optimisation process.

The performance of the proposed NN-based MPC is compared with two implementations of existing MPC: i) LMPC using a linear tyre model ii) NMPC using a nonlinear tyre model. These two models are commonly used for designing a dynamic model of a vehicle in the context of MPC, as discussed in section II. For both implementations, the same vehicle is controlled on various road conditions for different manoeuvrers. In addition, to evaluate the performance of the proposed controller in the presence of parametric uncertainty, tests are conducted for different parameter values of the system. The performance has been evaluated for the variation of two parameters of 1) road surface friction and 2) vehicle load (mass).

In this section, first, a description of the controlled vehicle and the corresponding simulated environment is briefly discussed. Then, the data collection process for training the NN-based model is reported. Finally, the formulation and performances of the MPC are reported.

A. SIMULATED VEHICLE

The simulated real vehicle controlled by the MPC has 14 DoF. This vehicle body has six DoF (longitudinal, lateral, vertical, yaw, pitch and roll) with four wheels, and each of them has two DoF (vertical and rolling). The vehicle body is connected to each wheel by a spring-damper suspension system. In addition, this model also includes a front-wheel-drive driveline, mapped spark-ignition engine, transmission, brake hydraulics and steering subsystems. This vehicle model is implemented in the MATLAB/Simulink environment. Fig. 5 shows the architecture of the simulated vehicle model. The nominal values for different parameters of this model are shown in Table 1.

TABLE 1. Nominal parameter of the simulated vehicle.

PARAMETERS	VALUE	UNIT
Vehicle Mass	1181	kg
Wheel Base	3.0750	m
Front axle from CG	1.51	m
Rear axle from CG	1.50	m
Height of CG	0.134	m
Steering Ratio	18	-
Track width	1.9220	m
Yaw moment of inertia	2066	kg m^{-2}

B. DATA COLLECTION

To implement the proposed offline NN-based MPC, first, the NN representing the dynamics of the vehicle needs to be trained. Data from a number of driving scenarios were collected from the simulated environment. During this process, the high fidelity model described in section V-A is used to drive the road on the road. A 3D simulation environment, 'Unreal Engine', is used for rendering the road environment. The Unreal Engine was interfaced with Simulink, which performs the vehicle dynamics operations. A Logitech G290 steering-pedal system is used to control the vehicle while the data is collected through communicating between the Unreal Engine and the vehicle dynamic model. Fig. 6 and 7 shows the architecture of the data collection system.

As the path profile has a significant effect on a vehicle's handling performance [53], for collecting data, the vehicle

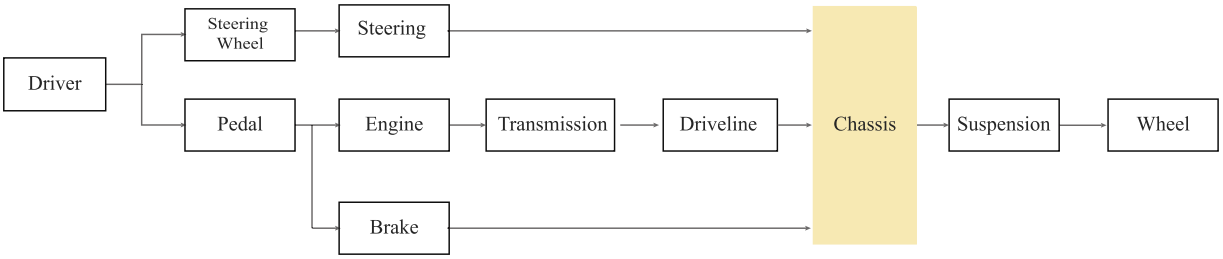


FIGURE 5. Architecture of the complex 14 DoF vehicle model.

FIGURE 6. System architecture for data collection.

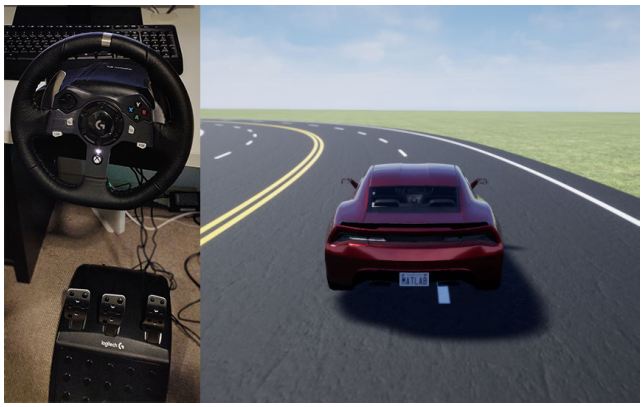


FIGURE 7. Driving controller and environment rendering in unreal engine for human demonstration.

was driven on three different road types numerous times, including 1) straight road (highway), 2) curved road (race-track), and 3) city block (a mixed of different turns). Different manoeuvres such as single lane and double lane change are also performed frequently for each road condition.

The ability of the controller is tested for different parameters variation. Data for different surface conditions such as dry road (friction coefficient, $\mu = 1$) and wet road ($\mu = 0.6$) was collected to achieve this. Moreover, data for different loading conditions were considered. To simulate this, the mass of the vehicle is varied due to the presence of a passenger on the vehicle. For no passenger condition, the nominal mass of the vehicle is considered. For the single passenger condition, 70kg is added to the mass of the vehicle. For the sake of simplicity, the additional mass is assumed to be distributed evenly on the vehicle.

For all driving tasks, at each time step, all vehicle states and inputs are recorded. The vehicle states include: current global position X and Y, yaw angle ψ, longitudinal velocity v_x, lateral velocity v_y, longitudinal acceleration a_x, lateral acceleration a_y, yaw velocity r and yaw acceleration $\dot{r}$. The corresponding steering wheel angle input δ_w and steering angle δ is also recorded. All data were collected at 100 Hz and then down-sampled at 33Hz.

C. TRAINING NEURAL NETWORK

1) OFFLINE TRAINING

Here, using the collected data, the NN is trained offline. The NN model represents the vehicle transition dynamics, so it can be used in the MPC to predict the future states of the vehicle. As the MPC is designed for the lateral control of the vehicle, the objective is to train a NN that estimate the transition of the vehicle states based on the history of the vehicle states and steering wheel angle input.

The gradient-based optimisation technique, 'Adam', is used for training the multilayer network. To this aim, the collected dataset is separated into three segments. Randomly chosen 70% of total data are used for training, 15% for validation, and the remainder 15% is used for testing. The Relu activation function is used for each hidden layer having $N = 100$ unit. The minibatch size of 50 is used with a learning rate of 0.001. The NN is trained for 10,000 iterations. A total of 230,000 trajectory steps (around 115 min of driving data) is used for training. The time required for training is 55 minutes on a computer with an Intel i7 processor and 32 GB RAM using MATLAB's Deep Learning Toolbox.

2) ONLINE TRAINING

In this approach, the network weight and bias values are initialised using the NW method [52]. To reduce the computational burden of online training, a smaller number of units for each hidden layer $N = 50$ is used. The network weights and biases are then updated using the gradient descent with momentum algorithm when a certain amount of data is available. Data is collected when the vehicle starts its operation. After collecting $N_s = 750$ steps of vehicle states and input data, the network weight and biases are updated. This process is repeated after $N_n = 50$ time steps. After each N_n time steps, new dataset is assembled which contains $N_s = N_c + N_n$ number of states and control sequence data from the vehicle. Here, N_c is the number of sequences from the old dataset. This process can be conducted in parallel with the MPC with

a separated processing core, so it is not considered a part of the real-time optimisation process of the MPC.

D. MPC FORMULATION

Using the offline trained NN-based vehicle transition model, the MPC controller is created based on the discussion in section IV. The following cost function is used for the MPC optimisation

$$J(\mathbf{x}_{k|t}, \mathbf{u}_{k|t}) = \sum_{k=0}^{N_p-1} w_d|\xi_d|^2 + w_\psi|\xi_\psi|^2 + w_\delta|\Delta\delta|^2 \quad (18a)$$

where,

$$\xi_d = \sqrt{\left(\hat{X}_{t+k|t} - X^{ref}_{t+k|t}\right)^2 + \left(\hat{Y}_{t+k|t} - Y^{ref}_{t+k|t}\right)^2} \quad (18b)$$

$$\xi_\psi = \sqrt{\left(\hat{\psi}_{t+k|t} - \psi^{ref}_{t+k|t}\right)^2} \quad (18c)$$

Here, the first term of the cost function ξ_d represents the distance error between the current position of the vehicle and the closest point of the reference path. $\hat{X}$ and $\hat{Y}$ is the predicted position of the vehicle using the transition model. Similarly, ξ_ψ expresses the angle error between the current yaw angle of the vehicle and the path angle in the global coordinate, where $\hat{\psi}$ is the estimated yaw angle of the vehicle. $w(\cdot)$ are the corresponding weight of each term. X^{ref}, Y^{ref} and ψ^{ref} represent the position and angle of the reference path to be followed by the vehicle. In addition, $\Delta\delta$ is the steering angle input rate used for lateral control of the vehicle.

The optimisation problem of the MPC is solved using the interior point optimisation method using the IPOPT package on a computer with an Intel i7 processor with multiple cores. The value of the prediction horizon, time step and weights of the cost function are listed in Table. 2.

TABLE 2. Value of controller parameters.

CONTROL PARAMETER	VALUE
Prediction horizon, N_p	8
Time step, Δt	0.033 s
Distance error weight, w_d	0.5
Heading error weight, w_ψ	10
Control input weight, w_δ	0.01

For the switched MPC approach with the online trained NN-model, the same cost function of (18) is used. However, the vehicle is started with the nonlinear dynamic model, and prediction performances for both nonlinear dynamic and adaptive NN-model are compared at each step. Prediction error for each model is calculated using the following equation

$$e_p(f^p, t) = \mathbf{w}_e|\mathbf{x}^p_t - \mathbf{x}^v_t|^2 \quad (19)$$

where, $\mathbf{x}^v = [X, Y, \psi, v_y, r]$ is the measured vehicle states, $\mathbf{x}^p$ is the predicted states of model f^p. To have an appropriate representation of each state, $\mathbf{w}_e = [1, 1, 1, 10, 10]$ is used. The NN-model is adapted at a regular interval when sufficient data is available, and the updated network is used for the prediction error calculation. Based on this prediction error, the switched MPC uses the NN-model when it is more accurate than the nonlinear dynamic model.

For both cases, MPC is only used for the lateral control of the vehicle and a separate longitudinal controller for the vehicle is designed. For the longitudinal control, a simple PI controller is used to maintain a predefined speed of the vehicle.

$$a_x = K_p(v_x - v_x^{ref}) + \int K_I(v_x - v_x^{ref}) \quad (20)$$

where, v_{ref} is the reference speed of the vehicle, K_P is the proportional and K_I is the integral gain.

E. PARAMETER TUNING

Parameter tuning of MPC weights plays an important role in the performance of the controller. MPC's performances can be improved by tuning the parameters even for specific road conditions and manoeuvres. To properly compare different MPCs, it is essential to provide a baseline approach for tuning its parameters. To be able to have a consistent comparison, the parameter for each controller is tuned using a Genetic Algorithm (GA)-based optimiser. The objective of the optimiser is to find proper tuning parameters with similar effort for each controller.

The following features are used to compare the performance of different the controller.

$$\text{maximum lateral error}: \ \xi_{d,max} = \max_{t\in[0,T]}|\xi_d(t)|$$

$$\text{maximum orientation error}: \ \xi_{\psi,max} = \max_{t\in[0,T]}|\xi_\psi(t)|$$

$$\text{average lateral error}: \ \xi_{d,rms} = \sqrt{\frac{1}{T}\int_0^T \xi_d(t)^2 dt}$$

$$\text{average orientation error}: \ \xi_{\psi,rms} = \sqrt{\frac{1}{T}\int_0^T \xi_\psi(t))^2 dt}$$

where, ξ_d is the lateral error and ξ_ψ is the orientation error. The optimal tuning parameters minimises the RMS and maximum tracking error. The following cost function is used for the GA optimisation:

$$J_{tune} = \xi_{d,rms} + \xi_{d,max} + \xi_{\psi,rms} + \xi_{\psi,max} \quad (21)$$

In the GA optimisation, a population size of 50 and a maximum number of generation of 100 are used.

Using this GA optimiser, a set of suitable parameters are chosen for proposed controllers. In addition, to remove the effect of the parameter tuning from the performance comparison, the same GA optimiser is used for the compared LMPC and NMPC.

VI. RESULTS AND DISCUSSION

In this section, the results of prediction performances of the designed vehicle models are shown. Moreover, the results of tracking performance of the proposed MPCs are presented. We also provide a detailed discussion on the results and corresponding comparisons with other controllers.

A. MODEL PERFORMANCES

Prediction performances of the offline trained NN-based vehicle model are evaluated, and a comparison with the observer vehicle data is shown in Fig 8. Here, the results for different road conditions are partitioned using the dotted vertical line. The first portion shows the prediction results for road surface friction of $\mu = 1$, and the second portion is for $\mu = 0.6$.

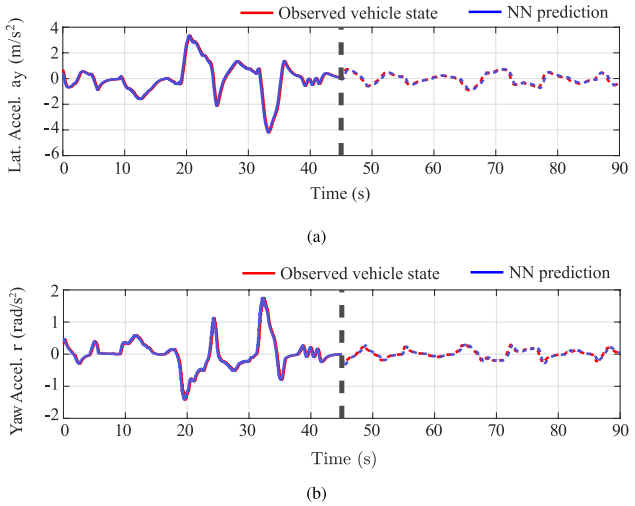


FIGURE 8. Performance comparison of observed data and trained NN's prediction. The variation in road-surface friction is indicated by the dotted line. In the first portion road surface friction coefficient $\mu = 1$ and in the second portion $\mu = 0.6$ is used.

The prediction performance of the model is also tested for different vehicle load conditions. Moreover, the model's prediction performance on roads with different curvatures for these parameter variations is recorded. The test road segments with different curvatures are shown in Fig. 9. Here, we can see that segment-2 has smaller curvatures than segment-1, whereas the curvatures for segment-3 is much higher than other segments. The performances of the model are tested for these three road segments under the aforementioned parameter variations. Here, we use the metric Root Mean Square Error (RMSE) to express the NN's performance. Figure 10 shows the RMSE for each output of the offline NN model using the test vehicle data and corresponding NN predictions. Figure 10a shows the RMSE of the lateral acceleration for different road segments under different parameter values. A similar result for the yaw acceleration output is shown in Fig. 10b.

In an MPC approach, the vehicle model is used to predict future vehicle states, which are then used to generate optimised control action. After applying the control action, the vehicle model is reinitialised using the state feedback. The estimation accuracy of the vehicle model up to a K-step ahead prediction horizon plays an important role in the performance of the MPC. Figure 11 show the K-step ahead prediction error for different vehicle models. For this comparison, the prediction horizon of MPC $K = 8$ is used. This means after

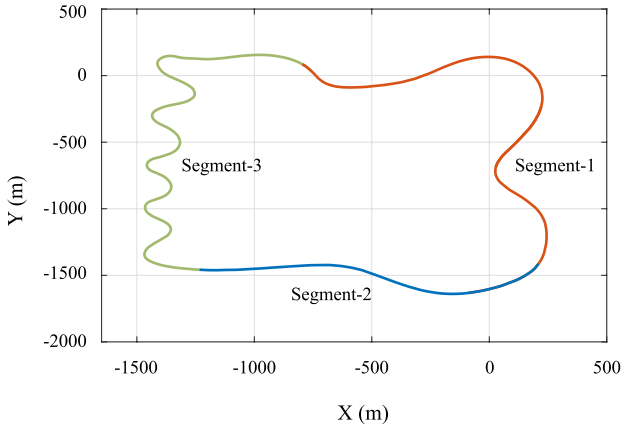


FIGURE 9. Test road segments with different curvatures.

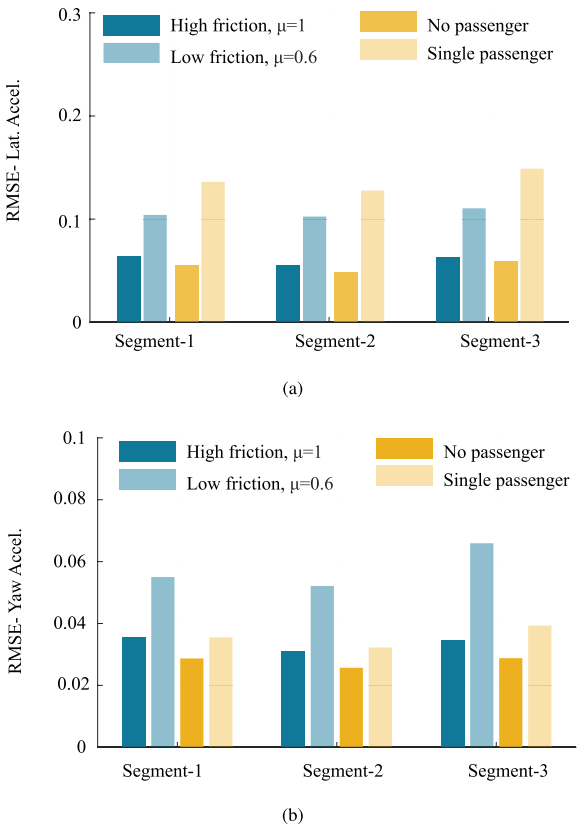


FIGURE 10. NN model performance for different road segments for parameter variations.

each K time steps, the states of the prediction model are updated using the state-feedback of the vehicle. The prediction error is calculated using (19).

Figure 11a shows the K-step ahead error comparison of two NN-based models and the nonlinear dynamic model for a constant steering angle operation of the vehicle. In addition, a similar comparison for an increasing steering angle operation of the vehicle is shown in Fig. 11b. In both figures, at each step, the mean error for the K-step prediction horizon is shown. The dotted vertical line represents the condition

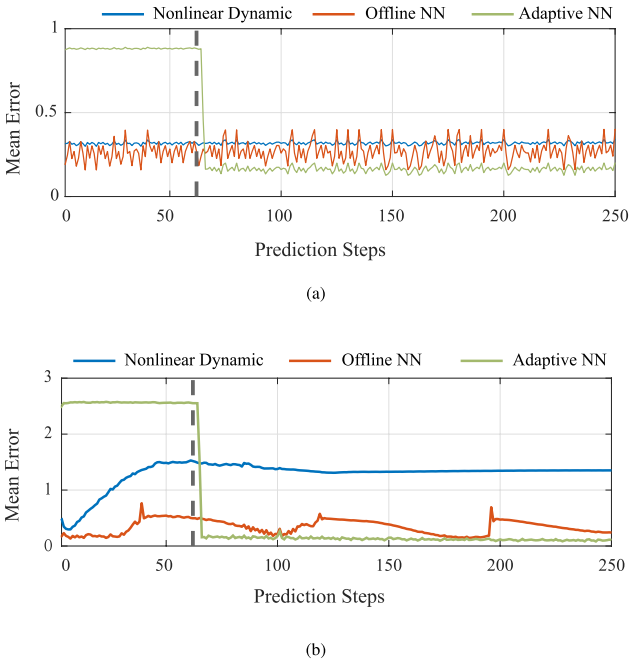


FIGURE 11. K-step ahead prediction error for vehicle operation with a) a constant steering angle b) increased steering. At each step, a mean prediction error for $K = 8$ prediction horizon are shown.

when the online NN has enough data and starts adapting the weights and biases of the NN. Here, the vehicle longitudinal speed is constant at 60 km h^{-1}.

From these results, it is apparent that the NN-based models reflect superior performances when properly trained. The online trained adaptive NN provides better performance when it has sufficient data and is trained for a number of iterations. This observation can be detected from the results at the beginning of the operation. However, this model eventually performs better than other models after a specific time. A switching MPC is proposed to circumvent this problem where the operation starts with a nonlinear dynamic model and switches to the NN model when it provides better performance.

B. TRACKING PERFORMANCE

1) MPC WITH OFFLINE NN-BASED MODEL

To test the proposed controller's performance with the offline NN-based model, two different manoeuvers are considered, 1) Single-Lane Change (SLC) and 2) Double-Lane Change (DLC). For both manoeuvres, the reference trajectory is collected from human driving with the same vehicle. Each manoeuvre is performed for the variation of two parameters: friction co-efficient and vehicle load. The controller's performance is compared with two conventional MPCs: LMPC and NMPC.

To evaluate the controller's efficacy with the parameter variations, the NN-based vehicle model is trained with the data from different variations of these parameters. For conventional MPCs, the vehicle models assume a constant value of these parameters, which is common in the literature of PTCs for AVs. Here, for the physical models, the friction coefficient is fixed at $\mu = 1$, and the vehicle load is at nominal vehicle load reported in Table 1. First, the vehicle is operated on two different road surface conditions and the results for each controller is recorded for both SLC and DLC manoeuvres. Fig 12 shows the trajectory and yaw angle

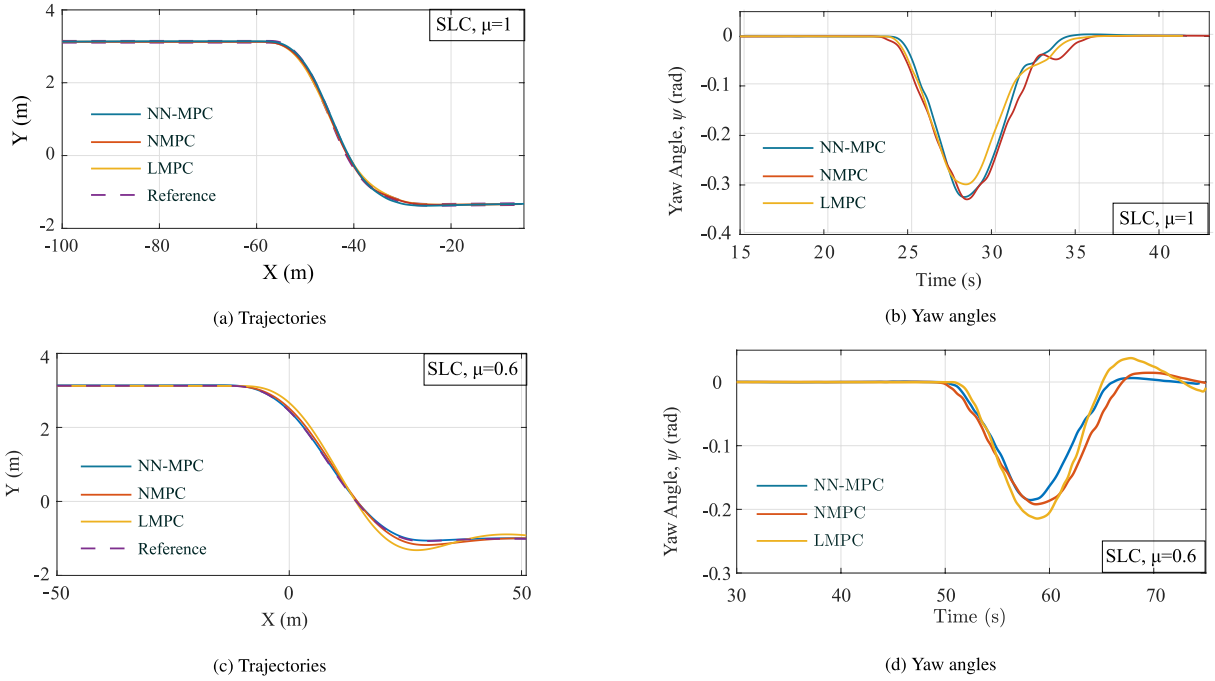


FIGURE 12. Tracking performance of the controllers for SLC manoeuvrer with a road surface friction coefficient of a-b) $\mu = 1$ and c-d) $\mu = 0.6$.

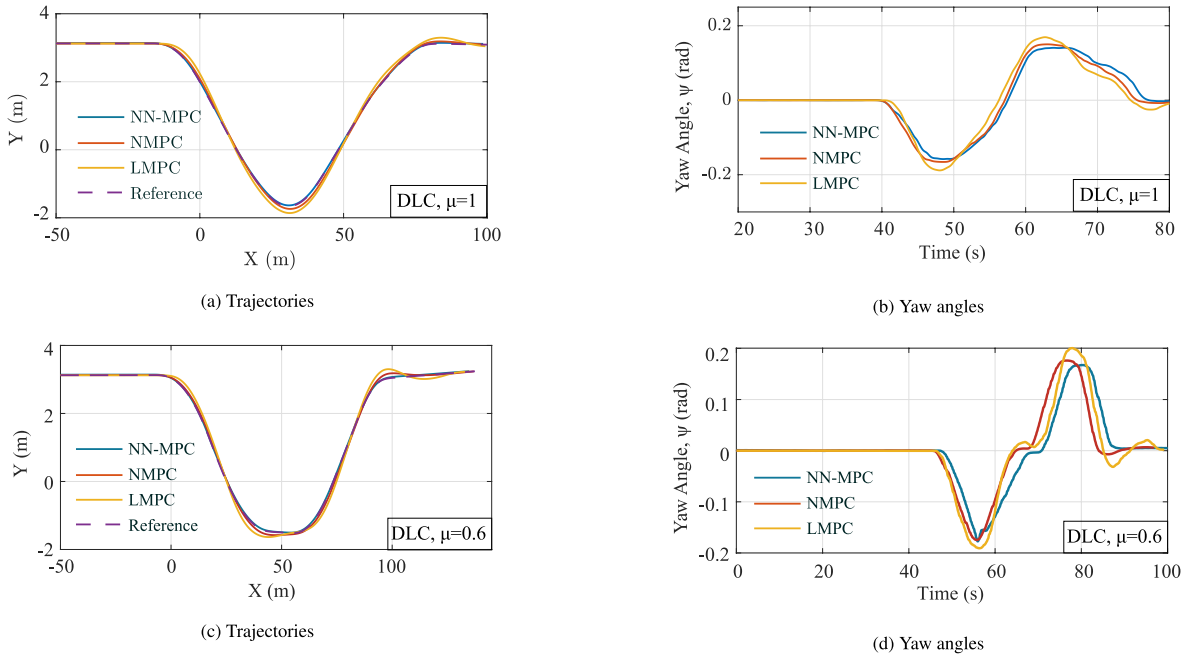


FIGURE 13. Tracking performance of the controllers for DLC manoeuvrer with a road surface friction coefficient of a-b) $\mu = 1$ and c-d) $\mu = 0.6$.

comparison of different controllers for two surface conditions ($\mu = 1$ and 0.6) for SLC maneuver. Similarly, for the DLC manoeuvre, the performance comparison for two different road surface conditions are shown in Fig 13. Figure. 14 shows the RMS error comparison for all these operations. For all operations, a forward velocity of 60 km h^{-1} is used. This vehicle speed is chosen based on the most common road speed limit information of Australia. According to the review report of the Victorian Government, Australia [54], 60 km h^{-1} is the most common speed limit for Australian roads with little to no pedestrian activities with a high number of access points.

The controller's performance is also evaluated for the variation of vehicle load. Load conditions are considered with no passenger and a single passenger. For the vehicle without passenger condition, the nominal mass of the vehicle reported in table 1 is used. For the single passenger condition, the average mass of a human 70 kg is added. Figure 15 shows the SLC manoeuvre for the single passenger condition for each controller. Similarly, Fig 16 shows the trajectories and yaw angles for the DLC manoeuvre. A forward velocity of 60 km h^{-1} is used for both cases. Finally, Fig. 17 shows the rms error comparison for no-passenger and single passenger condition.

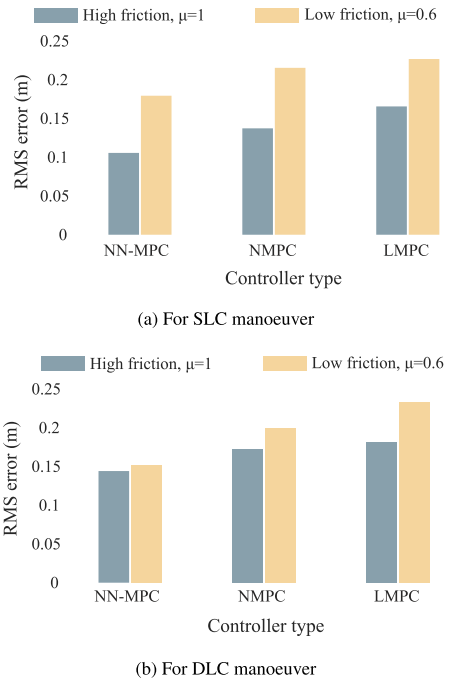
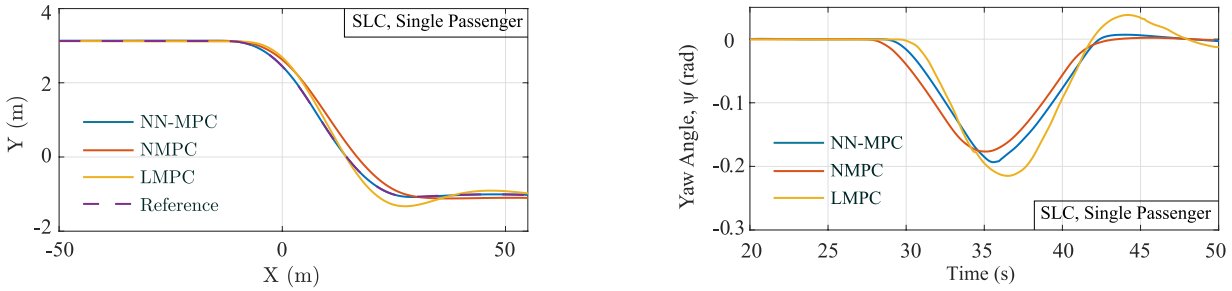


FIGURE 14. Tracking error comparison of the controllers for different surface friction co-efficient values for a) SLC and b) DLC manoeuver.

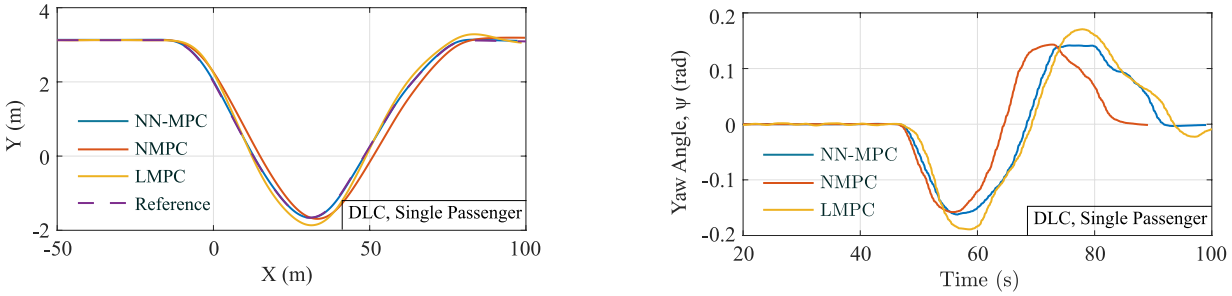
2) MPC WITH ONLINE NN-BASED MODEL

The performance of the online NN model-based switched MPC is discussed here. For the performance evaluation, the same road surface with a number of different manoeuvers is chosen. From the discussion in section VI-A as well as the prediction performance results shown in Fig. 11a and 11b, it is apparent that the adaptive NN approach requires a certain number of data and learning iterations to provide better performance than the physics-based dynamic model. To circumvent this problem, an SMPC is designed where the controller uses the nonlinear dynamic vehicle model until



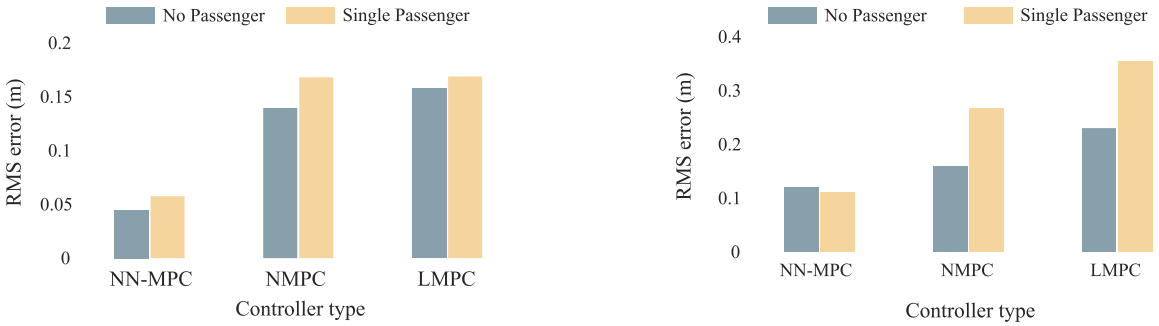
(a) Trajectories (b) Yaw angles

FIGURE 15. Tracking error comparison of the controllers for SLC manoeuver for single passenger load condition.



(a) Trajectories (b) Yaw angles

FIGURE 16. Tracking error comparison of the controllers for DLC manoeuver for single passenger load condition.



(a) SLC (b) DLC

FIGURE 17. Tracking error comparison of the controllers for vehicle load variation for a) SLC manoeuver and b) DLC manoeuver.

the NN-model provides better performance. For the clarity of presentation, we refer to the nonlinear dynamic model as the 'NL' model. Figure. 18 shows the trajectory generated by the SMPC. Here, the red portion of the trajectory is generated while using the NL model, whereas for the rest of the blue coloured trajectory, the online NN-model is used.

During the operation of the switched MPC, the weights and biases are adapted at a regular interval. This process can be conducted in parallel with the MPC with a separated processing core, so it is not considered a part of the real-time optimisation process of the MPC. Figure. 19 shows the prediction error comparison of NL and NN vehicle model calculated using (19) during the tracking task. For the clarity of presentation, the data is shown when the NN model has enough data and starts adapting the network.

The performance of the proposed SMPC is also compared with other controllers. The trajectories generated by different controllers are depicted in Fig. 20. The RMS error for each controller for the same tracking task is shown in Fig. 21.

C. DISCUSSION

From the observation of Fig.12-17, it is apparent that MPC with the offline trained NN model performs significantly better than the other two controllers even in the presence of parameter variations. Two important aspects to

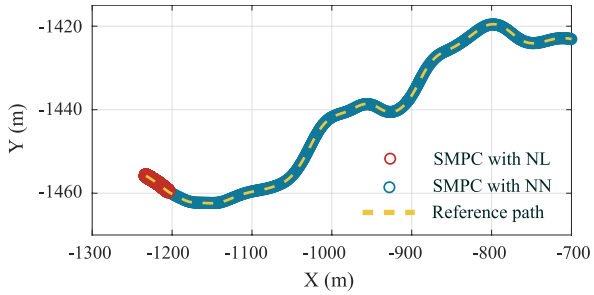


FIGURE 18. Trajectory generated by the proposed SMPC. Here, for generating the red portion of the trajectory, SMPC used the nonlinear dynamic model, and for the green portion of the trajectory adaptive NN model is used.

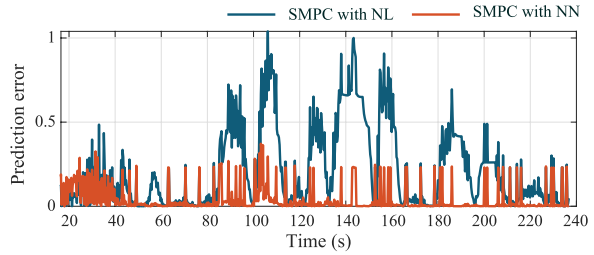


FIGURE 19. Prediction error comparison during SMPC tracking task. Here, prediction error for two available models for the SMPC during the tracking task is shown.

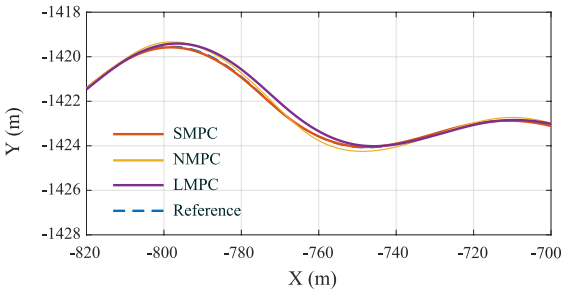


FIGURE 20. Trajectories generated by different controllers. Only a portion of the trajectory is shown for clarity.

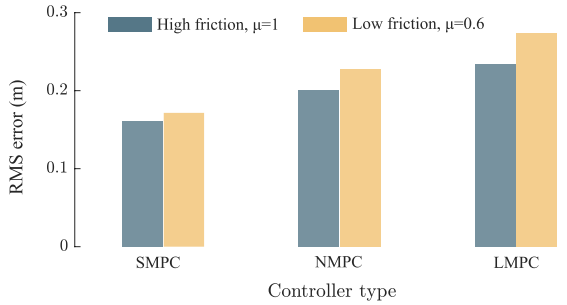


FIGURE 21. Tracking error comparison of the controllers for different surface friction co-efficient values.

note here: first, the performance of the NN-MPC is significantly better even when the NMPC and LMPC have a good approximation of the underlying parameter. This is because the NN-based vehicle transition model approximates the dynamics of the vehicle more comprehensively than the mathematical models. In addition, the performance of the NN-MPC does not degrade significantly (which happens to other controllers) due to the change in road-friction parameters. This is due to the fact that due to the used history of state and control input, the NN-based transition model can approximate latent states of the system without a significant increase in the computation cost.

The proposed NN-based approach is highly beneficial when the formulation of a mathematical model is complex. Based on the design and application, an AV can have different shapes and sizes. In addition, even the same type of vehicles are not identical and are guaranteed to have some degree of variation. Designing an analytical model for each of them is difficult and time-consuming. For most cases, the simplified analytical model may introduce uncertainties due to unmodelled dynamics. Using the data-driven approach to learn the vehicle dynamics, simplification is not required anymore, and the full range of dynamics can be identified and simulated. In this work, we only tested the performance of this approach for parametric uncertainties. However, this approach potentially can provide similar improved performances with the presence of nonparametric uncertainties, noise, and delay which are in the scope of our future works.

One of the important aspects of the proposed approach is that the NN-based MPC's efficacy depends on the size and quality of the dataset. A large amount of data from different road conditions are required for it to be highly efficient. Moreover, similar vehicles do not reflect the identical dynamic and can vary in different aspects. For example, some vehicle parameters change during the vehicle lifetime. A fixed vehicle model may not be an ideal solution in this case. To address this issue, the adaptive NN approach is proposed. In this case, the network weight and biases are adapted at a regular interval when a new set of data from the vehicle is available. This approach continuously changes the network based on the updated data. A mix of old and current information can be used for the adaptation, so the NN does not totally ignore the previous experiences when the new data is available. One of the bottlenecks of this approach is that the network needs a certain amount of data to be available before it provides proper results. An SMPC provides a good solution where the controller uses the nonlinear dynamic model when the prediction accuracy of the adaptive NN model is low.

Besides, from the results in Fig. 18-21, it is apparent that the proposed SMPC is capable of performing the optimal path tracking task. During the initial phase of the task, the controller uses the nonlinear dynamic model until the NN-model is ready. Then, the controller switches to the NN model. The performance of the proposed SMPC has been evaluated for different road surface friction conditions. From the tracking accuracy comparison shown in Fig. 21, it is clear that adaptive NN-based SMPC reflects significantly superior performance than the conventional MPCs.

Another important performance criterion of MPC is the computational cost. A complex model may increase accuracy; however, it may not be fast enough for real-time operation. From the results of Fig. 22, it is evident that

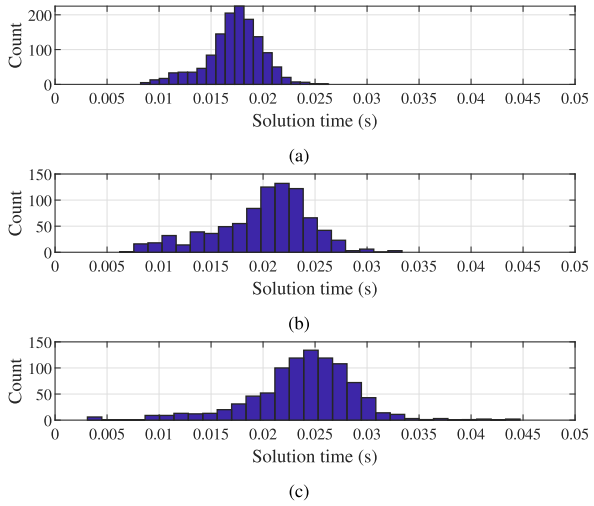


FIGURE 22. MPC optimisation solution time range for different controllers a) LMPC b) NMPC c) NN-MPC.

the proposed NN-based MPC is capable of real-time operation with a step size of $dt = 0.033$s. It provides a more accurate prediction performance without imposing a significant increase in the computational cost of online optimisation. Noting that the online NN can be operated in parallel with the MPC and is not considered a part of the MPC task.

For future work, the proposed approach will be implemented with more parameter variations and road conditions. In addition, this approach will be integrated with our previous works [55] on learning-based MPCs.

VII. CONCLUSION

Path Tracking Controller (PTC) is an integral subsystem of Autonomous Vehicles (AVs), responsible for controlling the vehicle on a predefined reference path. Different approaches have been proposed for designing PTCs; among them, Model Predictive Control (MPC) has shown to be a capable technique providing inherent robustness against uncertainties. However, an efficient MPC relies on the proper choice of the vehicle model used to predict future states. Moreover, a proper balance between complexity and accuracy is essential to have acceptable performance for the MPC. A simplified vehicle model can perform well on some operating conditions; however, due to unmodeled dynamics of the vehicle, MPC's performance may degrade for other conditions. It is noteworthy that a too complex model may not be suitable for the real-time optimisation requirement of the MPC.

Learning the vehicle dynamics from the vehicle operation data can provide a highly efficient alternative with a proper trade-off between accuracy and complexity. This works proposes learning the dynamics of a vehicle using a Neural Network (NN). An NN-based vehicle model approach has the potential to 1) provide a balanced performance in terms of accuracy and complexity, 2) reduce the effect of unmodelled dynamics by providing more accurate predictions without significantly increasing the complexity of the model, 3) accommodate approximation of nonlinearities of the model, 4) estimate latent system states, and 5) identify the system's internal representation of time-varying dynamics if properly trained with state and input history.

Here two approaches for MPC with an NN-based vehicle model have been proposed. In the first approach, the NN model was trained offline. The dataset for training was collected by driving the simulated vehicle on various road conditions and in the presence of parameter variations. In the second approach, an adaptive NN model was used where no data from the vehicle was required before starting the operation. It has been observed that this latter approach requires a certain amount of data and a number of adaptation iterations before it performs better than a conventional analytical nonlinear dynamic model. To circumvent this, a Switched MPC (SMPC) was designed to switch to NN-model when it outperforms the nonlinear dynamic model.

From the outcomes of the work, offline trained MPC outperforms the conventional MPC when the performance of the controllers are evaluated on different road conditions and in the presence of parameter variations. It is noted that this scheme requires a large dataset to be efficient in dynamic operating conditions. Furthermore, it has been observed that the SMPC with an adaptive NN-based model provides significantly superior performance compared to the proposed MPC with offline NN-model and the conventional MPC.

REFERENCES

[1] B. Paden, M. Cáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[2] J. M. Snider, "Automatic steering methods for autonomous automobile path tracking," Robot. Inst., Pittsburgh, PA, USA, Tech. Rep. CMU-RITR-09-08, 2009.

[3] F. Kuhne, W. F. Lages, and J. G. da Silva, Jr., "Model predictive control of a mobile robot using linearization," in *Proc. Mechatronics Robot.*, 2004, pp. 525–530.

[4] S. G. Vougioukas, "Reactive trajectory tracking for mobile robots based on non linear model predictive control," in *Proc. IEEE Int. Conf. Robot. Automat.*, Apr. 2007, pp. 3074–3079.

[5] I. Batkovic, M. Zanon, M. Ali, and P. Falcone, "Real-time constrained trajectory planning and vehicle control for proactive autonomous driving with road users," in *Proc. 18th Eur. Control Conf. (ECC)*, Jun. 2019, pp. 256–262.

[6] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Review and performance evaluation of path tracking controllers of autonomous vehicles," *IET Intell. Transp. Syst.*, vol. 15, no. 5, pp. 646–670, May 2021.

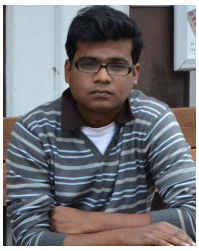
[7] S. F. Campbell, "Steering control of an autonomous ground vehicle with application to the DARPA urban challenge," Ph.D. dissertation, Dept. Mech. Eng., Massachusetts Inst. Technol., Cambridge, MA, USA, 2007. [Online]. Available: http://dspace.mit.edu/handle/1721.1/42301

[8] S. Thrun, M. Montemerlo, and H. Dahlkamp, "Stanley: The robot that won the DARPA grand challenge," in *Proc. Grand Challenge, Great Robot Race*. Berlin, Germany: Springer, 2007, pp. 1–43.

[9] E. Alcala, V. Puig, J. Quevedo, T. Escobet, and R. Comasolivas, "Autonomous vehicle control using a kinematic Lyapunov-based technique with LQR-LMI tuning," *Control Eng. Pract.*, vol. 73, pp. 1–12, Apr. 2018.

[10] A. De Luca, G. Oriolo, and C. Samson, "Feedback control of a nonholonomic car-like robot," in *Robot Motion Planning and Control* (Lecture Notes in Control and Information Sciences). Berlin, Germany: Springer, 1998, pp. 171–253.

[11] D. Chwa, "Sliding-mode tracking control of nonholonomic wheeled mobile robots in polar coordinates," *IEEE Trans. Control Syst. Technol.*, vol. 12, no. 4, pp. 637–644, Jul. 2004.
[12] L. Li, G. Jia, J. Chen, H. Zhu, D. Cao, and J. Song, "A novel vehicle dynamics stability control algorithm based on the hierarchical strategy with constrain of nonlinear tyre forces," *Veh. Syst. Dyn.*, vol. 53, no. 8, pp. 1093–1116, Aug. 2015.
[13] K. D. Do, Z. P. Jiang, and J. Pan, "Simultaneous tracking and stabilization of mobile robots: An adaptive approach," *IEEE Trans. Autom. Control*, vol. 49, no. 7, pp. 1147–1151, Jul. 2004.
[14] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, May 2007.
[15] P. Falcone, H. E. Tseng, F. Borrelli, J. Asgari, and D. Hrovat, "MPC-based yaw and lateral stabilisation via active front steering and braking," *Veh. Syst. Dyn.*, vol. 46, pp. 611–628, Sep. 2008.
[16] B. Gutjahr, L. Gröll, and M. Werling, "Lateral vehicle trajectory optimization using constrained linear time-varying MPC," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1586–1595, Jun. 2017.
[17] N. Mohajer, S. Nahavandi, H. Abdi, and Z. Najdovski, "Enhancing passenger comfort in autonomous vehicles through vehicle handling analysis and optimization," *IEEE Intell. Transp. Syst. Mag.*, vol. 13, no. 3, pp. 156–173, Oct. 2021.
[18] A. Katriniok and D. Abel, "LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics," in *Proc. 50th IEEE Conf. Decis. Control Eur. Control Conf. (CDC-ECC)*, Dec. 2011, pp. 6828–6833.
[19] A. Katriniok, J. P. Maschuw, F. Christen, L. Eckstein, and D. Abel, "Optimal vehicle dynamics control for combined longitudinal and lateral autonomous vehicle guidance," in *Proc. Eur. Control Conf. (ECC)*, Jul. 2013, pp. 974–979.
[20] M. Rokonuzzaman, N. Mohajer, and S. Nahavandi, "NMPC-based controller for autonomous vehicles considering handling performance," in *Proc. 7th Int. Conf. Control, Mechatronics Autom. (ICCMA)*, Nov. 2019, pp. 266–270.
[21] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "A multi-stage optimization formulation for MPC-based obstacle avoidance in autonomous vehicles using a LIDAR sensor," in *Proc. Dyn. Syst. Control Conf.*, vol. 2, Oct. 2014, pp. 1–4.
[22] D. Q. Mayne, E. C. Kerrigan, E. J. van Wyk, and P. Falugi, "Tube-based robust nonlinear model predictive control," *Int. J. Robust Nonlinear Control*, vol. 21, no. 11, pp. 1341–1353, 2011.
[23] S. Mata, A. Zubizarreta, and C. Pinto, "Robust tube-based model predictive control for lateral path tracking," *IEEE Trans. Intell. Vehicles*, vol. 4, no. 4, pp. 569–577, Dec. 2019.
[24] Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli, "A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles," *Veh. Syst. Dyn.*, vol. 52, no. 6, pp. 802–823, Apr. 2014.
[25] E. Kayacan, E. Kayacan, H. Ramon, and W. Saeys, "Robust tube-based decentralized nonlinear model predictive control of an autonomous tractor-trailer system," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 1, pp. 447–456, Feb. 2015.
[26] P. Hang, X. Xia, G. Chen, and X. Chen, "Active safety control of automated electric vehicles at driving limits: A tube-based MPC approach," *IEEE Trans. Transport. Electrific.*, early access, Jul. 28, 2021, doi: 10.1109/TTE.2021.3100843.
[27] O. Garcia, J. V. Ferreira, and A. M. Neto, "Design and simulation for path tracking control of a commercial vehicle using MPC," in *Proc. Joint Conf. Robot., SBR-LARS Robot. Symp. Robocontrol*, Oct. 2014, pp. 61–66.
[28] E. Kim, J. Kim, and M. Sunwoo, "Model predictive control strategy for smooth path tracking of autonomous vehicles with steering actuator dynamics," *Int. J. Automot. Technol.*, vol. 15, no. 7, pp. 1155–1164, Dec. 2014, doi: 10.1007/s12239-014-0120-9.
[29] I. Lenz, R. Knepper, and A. Saxena, "DeepMPC: Learning deep latent features for model predictive control," in *Proc. Robot., Sci. Syst.*, Jul. 2015, pp. 1–14.
[30] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin, "Learning quadrotor dynamics using neural network for flight control," 2016, *arXiv:1610.05863*. [Online]. Available: http://arxiv.org/abs/1610.05863
[31] A. Punjani and P. Abbeel, "Deep learning helicopter dynamics models," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Seattle, WA, USA, May 2015, pp. 3223–3230.
[32] M. K. Samal, S. Anavatti, and M. Garratt, "Neural network based system identification for autonomous flight of an eagle helicopter," *IFAC Proc. Volumes*, vol. 41, no. 2, pp. 7421–7426, 2008.
[33] V. A. Akpan and G. D. Hassapis, "Nonlinear model identification and adaptive model predictive control using neural networks," *ISA Trans.*, vol. 50, no. 2, pp. 177–194, Apr. 2011.
[34] P. W. J. van de Ven, T. A. Johansen, A. J. Sørensen, C. Flanagan, and D. Toal, "Neural network augmented identification of underwater vehicle models," *IFAC Proc. Volumes*, vol. 37, no. 10, pp. 263–268, Jul. 2004.
[35] Z. Yan and J. Wang, "Model predictive control for tracking of underactuated vessels based on recurrent neural networks," *IEEE J. Ocean. Eng.*, vol. 37, no. 4, pp. 717–726, Oct. 2012.
[36] G. M. Zeng, X. S. Qin, L. He, G. H. Huang, H. L. Liu, and Y. P. Lin, "A neural network predictive control system for paper mill wastewater treatment," *Eng. Appl. Artif. Intell.*, vol. 16, no. 2, pp. 121–129, Mar. 2003.
[37] S. Mohanty, "Artificial neural network based system identification and model predictive control of a flotation column," *J. Process Control*, vol. 19, no. 6, pp. 991–999, Jun. 2009.
[38] A. Grancharova, J. Kocijan, and T. A. Johansen, "Explicit output-feedback nonlinear predictive control based on black-box models," *Eng. Appl. Artif. Intell.*, vol. 24, no. 2, pp. 388–397, Mar. 2011.
[39] S. S. James, S. R. Anderson, and M. D. Lio, "Longitudinal vehicle dynamics: A comparison of physical and data-driven models under large-scale real-world driving conditions," *IEEE Access*, vol. 8, pp. 73714–73729, 2020.
[40] M. Da Lio, D. Bortoluzzi, and G. P. R. Papini, "Modelling longitudinal vehicle dynamics with neural networks," *Vehicle Syst. Dyn.*, vol. 58, no. 11, pp. 1675–1693, Nov. 2020.
[41] G. Garimella, J. Funke, C. Wang, and M. Kobilarov, "Neural network modeling for steering control of an autonomous vehicle," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 2609–2615.
[42] S. J. Rutherford and D. J. Cole, "Modelling nonlinear vehicle dynamics with neural networks," *Int. J. Vehicle Des.*, vol. 53, no. 4, p. 260, 2010.
[43] X. Ji, X. He, C. Lv, Y. Liu, and J. Wu, "Adaptive-neural-network-based robust lateral motion control for autonomous vehicle at driving limits," *Control Eng. Pract.*, vol. 76, pp. 41–53, Jul. 2018.
[44] H. Taghavifar and S. Rakheja, "Path-tracking of autonomous vehicles using a novel adaptive robust exponential-like-sliding-mode fuzzy type-2 neural network controller," *Mech. Syst. Signal Process.*, vol. 130, pp. 41–55, Sep. 2019.
[45] N. A. Spielberg, M. Brown, N. R. Kapania, J. C. Kegelman, and J. C. Gerdes, "Neural network vehicle models for high-performance automated driving," *Sci. Robot.*, vol. 4, no. 28, Mar. 2019, Art. no. eaaw1975.
[46] A. Nagariya and S. Saripalli, "An iterative LQR controller for off-road and on-road vehicles using a neural network dynamics model," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1740–1745.
[47] R. N. Jazar, *Vehicle Dynamics: Theory Application*. New York, NY, USA: Springer, 2014.
[48] R. Rajamani, "Lateral vehicle dynamics," in *Vehicle Dynamics Control*. Boston, MA, USA: Springer, 2012, pp. 15–46.
[49] H. B. Pacejka, *Tire Vehicle Dynamics*, 3rd ed. Oxford, U.K.: Butterworth-Heinemann, 2012.
[50] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision avoidance and stabilization for autonomous vehicles in emergency scenarios," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 4, pp. 1204–1216, Jul. 2017.
[51] M. Brown, J. Funke, S. Erlien, and J. C. Gerdes, "Safe driving envelopes for path tracking in autonomous vehicles," *Control Eng. Pract.*, vol. 61, pp. 307–316, Apr. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0967066116300831
[52] D. Nguyen and B. Widrow, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," in *Proc. IJCNN Int. Joint Conf. Neural Netw.*, Jun. 1990, pp. 21–26.
[53] N. Mohajer, M. Rokonuzzaman, D. Nahavandi, S. M. Salaken, Z. Najdovski, and S. Nahavandi, "Effects of road path profiles on autonomous vehicles' handling behaviour," in *Proc. IEEE Int. Syst. Conf. (SysCon)*, Apr. 2020, pp. 1–6.
[54] Vic Roads Australia. (Aug. 2012). *Victorian Speed Limit Review*. Accessed: Aug. 13, 2021. [Online]. Available: https://www.warrnambool.vic.gov.au/road-safety
[55] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Learning-based model predictive control for path tracking control of autonomous vehicle," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2020, pp. 2913–2918.



MOHAMMAD ROKONUZZAMAN received the B.Sc. degree in electrical and electronic engineering in 2009, and the M.Sc. degree in space science and technology with the specialization in space robotics and automation from Aalto University, Finland, in 2015. He is currently pursuing the Ph.D. degree with the Institute for Intelligent Systems Research and Innovation (IISRI), Deakin University, Australia. His research interests include control of the autonomous vehicle, human effects in autonomous driving, and learning-based control of autonomous and semi-autonomous systems.

NAVID MOHAJER received the B.Eng. degree in mechanical engineering and the M.Sc. degree in mechatronics engineering from the University of Tehran, Iran, in 2009 and 2012, respectively, and the Ph.D. degree in vehicle dynamics and mechanical engineering from Deakin University, Australia, in 2017. He is currently a Researcher with IISRI, Deakin University. His research interests include control and dynamic of autonomous vehicles, mechanical design and analysis of complex systems (kinematics and dynamics), and multibody systems (MBS).

SAEID NAHAVANDI (Fellow, IEEE) received the Ph.D. degree from Durham University, U.K., in 1991.

He is currently working as Alfred Deakin Professor, the Pro Vice-Chancellor, the Chair of engineering, and the Founding Director of the Institute for Intelligent Systems Research and Innovation, Deakin University. He has published over 1000 scientific papers in various international journals and conferences. His research interest includes modeling of complex systems, robotics, and haptics.

Prof. Nahavandi is a fellow of Engineers Australia (FIEAust), the Institution of Engineering and Technology (FIET), and the Australian Academy of Technology and Engineering (ATSE). He is the Editor-In-Chief of *IEEE Systems, Man, and Cybernetics Magazine*, a Senior Editor of IEEE Systems Journal and IEEE Access, and an Associate Editor of IEEE Transactions on Cybernetics.

SHADY MOHAMED received the B.Sc. and M.Sc. degrees in information technology from Cairo University, Giza, Egypt, in 2000 and 2003, respectively, and the Ph.D. degree in control theory from Deakin University, Geelong, VIC, Australia, in 2009. He is currently an Associate Professor with the Institute for Intelligent Systems Research and Innovation (IISRI), Deakin University. His research interests include interdisciplinary research involving signal processing, control theory, human biodynamics, haptics, and medical imaging.