Transportation Research Part C 156 (2023) 104329

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Real-time forecasting of driver-vehicle dynamics on 3D roads: A deep-learning framework leveraging Bayesian optimisation

Luca Paparusso *, Stefano Melzi, Francesco Braghin

Department of Mechanical Engineering, Politecnico di Milano, Via Privata Giuseppe La Masa, 1, 20156 Milano MI, Italy

ARTICLE INFO

Keywords:
Trajectory forecasting
Motion cueing
Recurrent neural network
Driver-vehicle dynamics
Road geometry
Bayesian optimisation

ABSTRACT

Most state-of-the-art works in trajectory forecasting for automotive target predicting the pose and orientation of the agents in the scene. This represents a particularly useful problem, for instance in autonomous driving, but it does not cover a spectrum of applications in control and simulation that require information on vehicle dynamics features other than pose and orientation. Also, multi-step dynamic simulation of complex multibody models does not seem to be a viable solution for real-time long-term prediction, due to the high computational time required. To bridge this gap, we present a deep-learning framework to model and predict the evolution of the coupled driver-vehicle system dynamics jointly on a complex road geometry. It consists of two components. The first, a neural network predictor, is based on Long Short-Term Memory autoencoders and fuses the information on the road geometry and the past driver-vehicle system dynamics to produce context-aware predictions. The second, a Bayesian optimiser, is proposed to tune some significant hyperparameters of the network. These govern the network complexity, as well as the features importance. The result is a self-tunable framework with real-time applicability, which allows the user to specify the features of interest. The approach has been validated with a case study centred on motion cueing algorithms, using a dataset collected during test sessions of a non-professional driver on a dynamic driving simulator. A 3D track with complex geometry has been employed as driving environment to render the prediction task challenging. Finally, the robustness of the neural network to changes in the driver and track was investigated to set guidelines for future works.

1. Introduction

The field of control systems has considerably benefited from the advances in deep learning obtained in the last decade. The reason is to be found in the excellent results achieved by deep-learning algorithms in modelling complex dynamical systems (Kutz, 2017; Yeung et al., 2019). Particularly, modern control applications as autonomous driving and robot navigation, which deal with a dynamic and varied environment, need to manage more information to properly model the whole dynamics of the scene, including the behaviour of other autonomous agents or the interactions between humans and machines. As a consequence, the resulting dynamics becomes rather articulated, and in many cases it is even arduous to interpret the underlying physics that governs the system evolution. Under these conditions, data-driven models, especially deep-learning frameworks, have been demonstrated to embody a viable and efficient alternative to classical first-principle models.

Trajectory forecasting algorithms constitute one of the most important applications of deep learning in the area of control systems. They generate predictions on the future motion of the agents in the control scenario, which can be used to enhance the

* Corresponding author.
E-mail address: luca.paparusso@polimi.it (L. Paparusso).

https://doi.org/10.1016/j.trc.2023.104329
Received 6 May 2022; Received in revised form 24 July 2023; Accepted 3 September 2023
Available online 21 September 2023

promptness and performance of the controlled system. However, the majority of the state-of-the-art works in trajectory forecasting target predicting the future pose of the agents in the scene, that is their position and orientation in space. This is particularly useful when the core task of the downstream planning and control stacks is pose dependent only, for instance collision avoidance tasks.

On the other hand, there exist many automotive applications for which pose prediction of the vehicle-driver system is not sufficient to design proper simulation and control frameworks. As two motivating examples, predicting a more varied set of vehicle signals, as accelerations and rotational velocities, may significantly impact on motion cueing algorithms for dynamic driving simulators (Dagdelen et al., 2009) and emergency forecasting for shared autonomy (Schwarting et al., 2018). Also, even if the driver commands are known, simulating the vehicle dynamics with accurate multibody models is computationally expensive, so that long-term forecasts are not real-time obtainable. The extension of state-of-the-art trajectory forecasting algorithms to cope with a more varied set of signals is not trivial. In fact, it involves modelling and forecasting the dynamics of the coupled system constituted by a human driver and the vehicle at once, which has been a core problem in automotive research for half a century (Cacciabue, 2007; Plöchl and Edelmann, 2007). Supported by the previous motivating examples, in this work we aim to extend trajectory forecasting methods to design a simulation/prediction framework that is

- capable to model and predict the main dynamics of the driver-vehicle system on a complex 3D track at once, that is forecasting the driver behaviour and modelling the vehicle response;
- *data-driven*, to be tailored on a specific human driver;
- *road-geometry dependent*, to require few road geometry examples during training for good generalisation on new roads;
- *computationally efficient*, to be employed in simulation and for online control tasks;
- *self-tunable* during training, to adapt to specific applications and user-demanded output signals.

Statement of contributions. To bridge this gap, we propose a deep-learning framework based on LSTM autoencoders and leveraging Bayesian Optimisation, to predict a set of signals describing the driver-vehicle system dynamics. We call our framework *DriVe-forecast*. The LSTM encoder–decoder structure is justified by many state-of-the-art works in trajectory forecasting. The following contributions are then added to cope with the above-stated requirements.

1. The framework is designed to be general purpose, so that the user can specify the dynamics of interest. Without losing generality, we propose a basic set of features that can be measured/estimated and used for common control applications, and that well represent the main driver-vehicle dynamics.
2. Instead of splitting the problem into two parts, namely driver's intent prediction and vehicle dynamics simulation, the driver-vehicle dynamics is predicted at once. This allows to consider a coarser prediction horizon discretisation, which makes the computational time suited for real-time application. By splitting the problem instead, a reliable vehicle dynamics simulation would require a finer prediction horizon for the driver's intent, as well as a slower dynamics propagation by means of the multibody model.
3. The network fuses information on the realised past with the road geometry that the driver is about to travel, to generate context-aware predictions of the desired features for a predefined number of future time steps. Differently from state-of-the-art methods, we consider a complex 3D road instead of a urban structured road (e.g. intersection, crossing) in the formulation. This is done to model race tracks and suburban roads, in view of the applications mentioned in the previous section. However, we remark that the problem formulation can be integrated to support urban structured roads.
4. We introduce a parametric loss function to train the neural network. It allows to control the relative importance between forecasting errors on primary features, i.e. the ones of interest, and on secondary features, i.e. the auxiliary ones employed to help the network learning the dynamics of the system. Resorting to Bayesian optimisation, the loss function hyperparameters and the number of trainable parameters of the network are tuned to achieve a suitable trade-off between forecasting performance and computational time at the test phase. The latter is in fact fundamental in online implementations for control purposes.
5. The proposed approach is validated through a case study on motion cueing. An experimental campaign is conducted on a dynamic driving simulator to collect the dataset, train and validate the neural network. The dataset has been composed as follows.

    - It refers to laps of a specific non-professional driver on a specific track. The driver is non-professional, so that each lap is different from previous ones. This makes the prediction problem more challenging, even considering testing on the same driver and track;
    - we also include in the test portion of the dataset some laps performed by a different driver or on a different track to show the generalisation properties of the network, despite the constant driver and track kept during training.

    The scope of this work is not to capture the behaviour of different drivers, but to link the predicted driver/vehicle dynamics (not only trajectories) to the road information, with limited computational burden.

A schematic representation of the proposed prediction framework is depicted in Fig. 1.

Sections organisation. The remainder of this paper is organised as follows. Related works on trajectory forecasting, vehicle dynamics modelling and motion cueing are presented in Section 2. The prediction problem is formulated in Section 3. In Section 4, the proposed deep-learning architecture to solve the prediction problem is presented. The application of our method to the case study, centred on motion cueing algorithms, and the corresponding results are reported in Section 5; in the same section, the

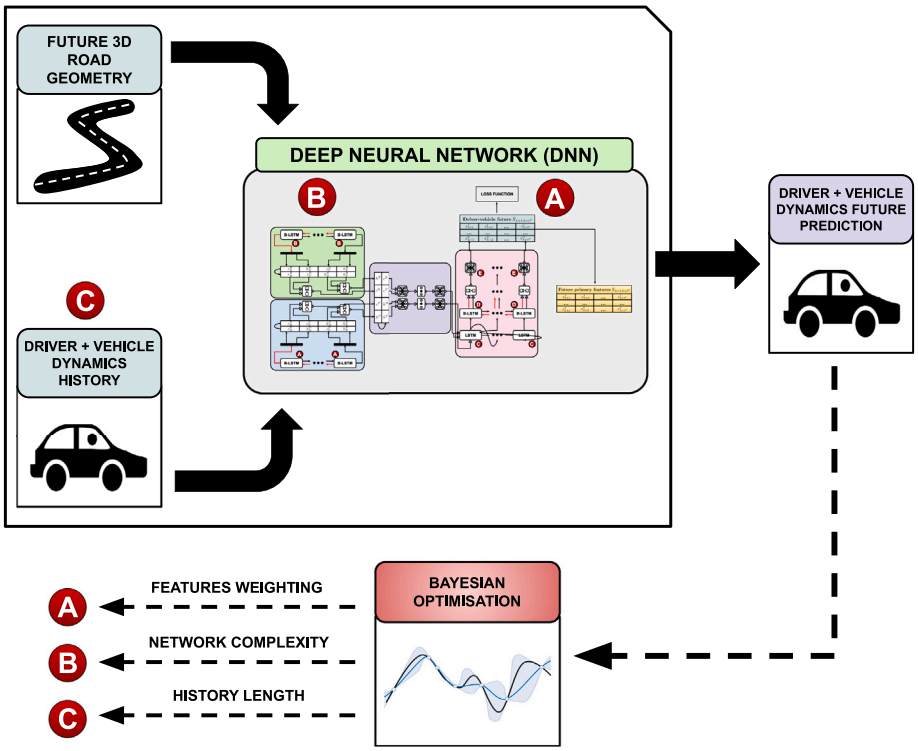


Fig. 1. Schematic representation of the proposed method. A deep neural network is used to forecast the future dynamics of the driver-vehicle system according to user-specified preferences. A 3D road geometry is considered to render the framework context-aware, guaranteeing generalisation on new roads and drivers with few training data. Finally, a Bayesian optimiser monitors the neural network outputs and tunes the prediction framework to guarantee good prediction performance as well as feasible computational times for online applicability.

dataset preparation and the hyperparameters tuning procedure are extensively described. In Section 6, we further investigate the generalisation properties of the proposed method, by testing it on data obtained with a different driver and a different track. Finally, the conclusions of this work and potential future studies are presented in Section 7.

2. Related work

2.1. Trajectory forecasting methods

Predicting the future spatial location of agents in urban or structured road with deep learning has been an active research field in the recent years (Mozaffari et al., 2020; Rudenko et al., 2020). In Morton et al. (2017), Long Short-Term Memory (LSTM) neural networks are used to predict the acceleration distributions of cars on highway, modelling in-lane motion only. LSTM networks with self-attention mechanisms are also used in Sun and Kim (2021) to forecast the next location and travel time of a vehicle in a city-wide environment. A probabilistic deep neural network is designed in Zyner et al. (2020) to predict distributions of vehicles trajectories in a roundabout, i.e. in-plane positions at any time instant, assuming that each vehicle can be modelled through its in-plane position, heading and longitudinal velocity. In Wang et al. (2018), Zhang et al. (2020), motion recognition and trajectory forecasting algorithms are applied to human–robot collaboration tasks, in which collision avoidance and safety guarantees constitute a strict requirement. Convolutional Neural Networks (CNN) are employed in Dominguez-Sanchez et al. (2017) to predict the motion of pedestrians for autonomous driving applications.

Recent trends in multi-agent trajectory forecasting are geared towards conditioning the prediction of the surrounding agents pose on the ego-vehicle controls, to produce interaction-aware forecasts (Salzmann et al., 2020). A different approach to trajectory forecasting for control tasks, presented in Ivanovic et al. (2020), consists in producing interaction-aware state-space equation models instead of tracklets as outputs of the network, which can be directly employed for planning and control tasks.

2.2. Vehicle dynamics modelling

The dynamics of the vehicle, in terms of response of its constitutive elements to known driving inputs, e.g. throttle, was widely investigated in the past (Gillespie, 1992; Milliken and Milliken, 1994; Blundell and Harty, 2004). However, modelling the decision-making process of a human driver, which lead to execute control actions considering environment, still represents an open research

field. Unlike autonomous navigation systems, a human driver shows a varied, stochastic and, most of the times, non-optimal behaviour. Moreover, drivers have personal driving styles, which are strictly related to their own experience.

Many works have tried to emulate human drivers using control systems, e.g. Braghin et al. (2008), with simplified models for the vehicle dynamics. These architectures allow to accurately describe the behaviour of professional drivers on track, whose aim is to minimise the lap time, but it could be difficult or even impossible to tune them so as to replicate a non-professional driver. To take into account the complex coupling effects between the driver and vehicle systems, and to capture the human nature of the driver, data-driven approaches may be employed to model and forecast the evolution of the whole system. Also, the high modelling performance of these methods allows to investigate how the geometry of the road influences the future driver/vehicle dynamics, which is considered in our work.

2.3. Road geometry modelling

Modelling the road is crucial in automotive applications such as trajectory forecasting and autonomous navigation. Map information is typically included in the problem in two ways:

- using implicit representations, i.e. direct sensor acquisitions without semantic information (Xiao et al., 2022), such as LIDAR point clouds (Casas et al., 2021) or bird-eye-view images (Walker et al., 2016). In this case, the road information is fused with the traffic agents' information (e.g. video of an intersection). These representations naturally result into voxelized 3D tensors, which are particularly suited to be directly processed by Convolutional Neural Networks (CNN) (Casas et al., 2021);
- using explicit information about the road structure, based on High-Definition (HD) Maps (Liu et al., 2020). HD maps are detailed representations of the road network, which include lane boundaries locations, traffic signs and drivable areas information. These maps are typically created by postprocessing several sensor data (Li et al., 2022; Bao et al., 2023; Jiao, 2018), such as LIDAR and GPS. From HD maps, it possible to extract sub-representations, such as Vector Maps, Occupancy Grid Maps and Topological Maps (Scheel et al., 2021; Suo et al., 2021; Deo et al., 2022).

All the previous representations are useful in autonomous driving tasks, where the focus is on motion observed from bird-eye view. Our goal is rather to model the vehicle dynamics, in and out of plane, considering a 3D road geometry. For this reason, we must include a more traditional road representation in our pipeline, based on differential geometry of curves.

2.4. Motion cueing

We also provide references to motion cueing, as it embodies one of the motivating examples of this work, and core of the case study for the evaluation of the performance of our method.

Motion cueing is a planning algorithm that provides motion references to the dynamic platform of driving simulators (Asadi et al., 2017; Mohammadi et al., 2019). The objective of the planning problem is to reproduce the translational accelerations and rotational velocities at the driver's head that would be generated on a real vehicle. In fact, it is through these quantities that humans perceive motion, thanks to their vestibular system. Modern implementations of motion cueing algorithms are based on constrained optimal control, to account for workspace limitations. The most used technique is MPC (Beghi et al., 2012, 2013; Bruschetta et al., 2019), whose formulation includes the system and reference evolution in the future (predictive horizon) to generate an optimised and prompt motion, anticipating future situations. However, the state-of-the-art solutions developed so far do not fully exploit the potential benefits introduced by MPC. To avoid predicting the future behaviour of the driver, they employ a constant reference throughout the predictive horizon, equal to the one that is measured in simulation at the time of execution. As shown in Grottoli et al. (2018), knowing the reference future evolution in advance and extending the predictive horizon would allow to outperform the state-of-the-art results. This represents a motivating example for our work.

3. Problem formulation

The formulation of the prediction problem shall be based on the emulation of the decision-making process of a human driver. At any time instant, drivers are aware of the vehicle state and their previous driving actions, which carry information about the recent past context that they have operated in. However, the past only does not provide a complete picture to characterise the resulting motion. In fact, drivers are highly conditioned by the road geometry that they see ahead, and act accordingly. Therefore, the proposed formulation includes information on the road geometry into the prediction model, so as to highlight the dependency between driving actions and environment properties.

The previous statements are now formalised in mathematical notation. Let us consider the current time step $k \in \mathbb{N}$. Given the 3D geometry of the track, that is the spatial location of the centreline and the road margins; given matrix $X_{(k-t^P:k)}$ describing the values assumed by $n \in \mathbb{N}$ vehicle features monitored in the last $t^P + 1$ time steps, where $t^P \in \mathbb{N}$; the objective is to predict matrix

$X_{(k+1:k+t^F)}$, describing the values assumed by $q \in \mathbb{N}$ vehicle features in the future t^F time steps, where $t^F \in \mathbb{N}$, $t^F \geq 1$. The q features are a subset of the n features monitored in the past, thus $q \leq n$. The matrices $X_{(k-t^P:k)}$ and $X_{(k+1:k+t^F)}$ are defined as

$$\begin{aligned} X_{(k-t^P:k)} &:= \begin{bmatrix} x^1_{k-t^P} & x^2_{k-t^P} & \dots & x^n_{k-t^P} \\ x^1_{k-t^P+1} & x^2_{k-t^P+1} & \dots & x^n_{k-t^P+1} \\ \dots & \dots & \dots & \\ x^1_k & x^2_k & \dots & x^n_k \end{bmatrix}, \\ X_{(k+1:k+t^F)} &:= \begin{bmatrix} x^1_{k+1} & x^2_{k+1} & \dots & x^q_{k+1} \\ x^1_{k+2} & x^2_{k+2} & \dots & x^q_{k+2} \\ \dots & \dots & \dots & \\ x^1_{k+t^F} & x^2_{k+t^F} & \dots & x^q_{k+t^F} \end{bmatrix}, \end{aligned} \tag{1}$$

where x^j_i indicates the value of the jth vehicle feature at the ith time step. The 3D geometry of the track will be used to extract a set of $m \in \mathbb{N}$ road geometry features.

In the next sections, We detail the choice of the two main ingredients of the prediction framework, that is

- a basic set of features describing the main driver-vehicle dynamics;
- a set of features encoding the future road geometry.

3.1. Definition of the road geometry

In prediction and control applications, it is fundamental to describe the geometrical properties of environment with features characterised by a sufficiently smooth evolution over time. To model the road and the vehicle position along the 3D road, we refer to the global coordinates of the left and right road margins, the road centreline and the vehicle centre of gravity (CG) with respect to a global orthonormal reference frame $X^G - Y^G - Z^G$. The axis Z^G points in the opposite direction with respect to the gravity vector. We aim to derive parametric curves describing the margins and the centreline using 5th-order-polynomial splines, being the curvilinear abscissa the independent variable of each curve. Starting from the sampled global coordinates of the road margins, the splines knots are placed using piecewise linear interpolation and resampling with a coarser spacing. Then, the splines parameters are computed by minimising the splines normed distance from the original sampled points. The employment of 5th-order polynomials ensures the continuity of the splines up to the second derivative.

Keeping the centreline curvilinear abscissa as the independent variable and using the splines, it is possible to derive the following road geometry features:

- the road width;
- the first derivative of the coordinate in Z_G with respect to the centreline curvilinear abscissa, i.e. the pitch slope;
- the road bank angle, i.e. the lateral slope;
- the centreline signed curvature in the $X^G - Y^G$ plane;
- the second derivative of the coordinate in Z_G with respect to the centreline curvilinear abscissa.

The values of these features carry information on the context in which the vehicle operates. The road width influences the lateral distance with respect to the centreline maintained by the driver along the track; the road pitch and lateral slopes play a key role in the dynamics of the vehicle, as they influence the orientation of the gravity vector with respect to the vehicle principal axes; finally, the centreline signed curvature in the $X^G - Y^G$ plane and the second derivative of Z^G with respect to the centreline curvilinear abscissa provide information on the shape of the track, allowing to distinguish different types of turn and straightaway.

The global coordinates of the centreline are also employed as references to define the relative distance and relative yaw of the vehicle, which are included among the vehicle features, as described in the next section.

3.2. Definition of the vehicle features

As specified above, the q vehicle features that we aim to forecast constitute a subset of the full set of n vehicle features considered in the method. The full set is fundamental to help the data-driven method learning the dynamics of the underlying physical process that we want to describe. Moreover, this becomes even more important considering that the driver could potentially lead the vehicle to very different situations on the 3D road, so that a data-driven algorithm failing to understand the driver-vehicle system dynamics would not be able to correctly generalise on new data. In view of these considerations, we will show in Section 4 that the proposed data-driven scheme is designed to predict all of the n features, while it adopts a weighting strategy to maximise the prediction performance on the q desired features.

The selected n vehicle features are:

- the vehicle CG accelerations with respect to its principal axes;
- the vehicle chassis angular velocities with respect to its principal axes;
- the vehicle CG relative distance and vehicle chassis relative yaw with respect to the centreline;

- the vehicle CG velocities with respect to its principal axes;
- the throttle percentage, brake percentage, steering angle, steering angle rate and gear, i.e. driver commands.

In this case, the number of vehicle features is $n = 16$. It is necessary to notice that the choice of the vehicle features depends on the level of detail in the description of the vehicle dynamics that one is interested to model. In this work, we decide to model the part of the driver-vehicle system dynamics constituted by those features that can be measured/estimated and used for common control applications, e.g. ADAS. However, the framework is designed to be general, so that the user can include other features of interest into the problem. The choice was also supported by a correlation analysis.

4. Proposed prediction strategy

The proposed prediction strategy is detailed in this section. The matrix $X_{(k-t^P:k)}$, introduced in Section 3, will be referred to as matrix of the past from now on. It contains the information on the recent past that the system driver-vehicle has just experienced. As stated above, the past is not sufficient to fully characterise the behaviour of the system in the future, since it is highly correlated with the geometrical context that is encountered. To introduce the information on the road geometry that the driver sees ahead, we consider the curvilinear abscissa s_k of point P_k, defined as the point of the centreline that is closest to the vehicle CG at the time step k. Considering that at the current time step the road geometry influencing the finite horizon prediction depends on a limited portion of the whole track, we define the maximum distance that the driver can see ahead $d^R \in \mathbb{R}$, $d^R \geq 0$, and the number of spatially equidistant discretisation points $p^R \in \mathbb{N}$ in the continuous interval $(s_k, s_k + d^R]$. From the corresponding values of curvilinear abscissa, denoted as $s_{k|1}, s_{k|2}, \dots, s_{k|p^R}$, the road geometry features are evaluated and stored into matrix $\bar{X}_k^R$. Therefore, its structure is

$$\bar{X}_k^R := \begin{bmatrix} x^1_{k|1} & x^2_{k|1} & \dots & x^m_{k|1} \\ x^1_{k|2} & x^2_{k|2} & \dots & x^m_{k|2} \\ \dots & \dots & \dots & \\ x^1_{k|p^R} & x^2_{k|p^R} & \dots & x^m_{k|p^R} \end{bmatrix}, \tag{2}$$

where $x^j_{k|i}$ indicates the value of the jth road feature evaluated at the curvilinear abscissa $s_{k|i}$.

Finally, following the considerations introduced in Section 3.2, we introduce matrix $\bar{X}_{(k+1:k+t^F)}$ expressing the full set of n vehicle features in the future t^F time steps, as

$$\bar{X}_{(k+1:k+t^F)} := \begin{bmatrix} x^1_{k+1} & x^2_{k+1} & \dots & x^n_{k+1} \\ x^1_{k+2} & x^2_{k+2} & \dots & x^n_{k+2} \\ \dots & \dots & \dots & \\ x^1_{k+t^F} & x^2_{k+t^F} & \dots & x^n_{k+t^F} \end{bmatrix}. \tag{3}$$

We remark that matrix $X_{(k+1:k+t^F)}$, i.e. the goal of our problem, can be obtained by extracting the subset of q features of interest from matrix $\bar{X}_{(k+1:k+t^F)}$, being $q \leq n$.

4.1. Deep neural network

To solve the prediction problem formulated above, we propose a deep-learning architecture. The structure of the artificial neural network, exploiting an encoder–decoder topology to synthesise the information of the past $X_{(k-t^P:k)}$ and the road seen ahead $\bar{X}_k^R$, is now presented.

The scheme of the network is shown in Fig. 2. The matrix of the past $X_{(k-t^P:k)}$ is encoded using a bidirectional Long-Short-Term-Memory (LSTM) layer with u_e hidden units, for their high performance in sequence encoding (Britz et al., 2017). The last hidden states h^P_F, h^P_B, and cell states c^P_F, c^P_B of the LSTM layers in the forward and backward directions are merged together through concatenation, to form one hidden states vector h^P and one cell states vector c^P. The road geometry matrix $\bar{X}_k^R$ is encoded by means of another bidirectional LSTM layer with u_e hidden units, employing the same merge mechanism, to obtain the hidden states vector h^R and cell states vector c^R. The hidden states and cell states of the two encoders are then concatenated, to create an extended hidden states vector h and an extended cell states vector c. Two dense layers with $2u_e$ and u_d neurons are applied to both h and c without weights sharing, to merge the information carried by the encodings of the past and the geometry of the road in the future, and to uniform them to the dimensions of the decoder. A dropout layer with rate r is placed between each couple of dense layers to promote regularisation. Consequently, the vectors h^D and c^D are generated.

The decoder consists of two stages. The first is a unidirectional LSTM layer with u_d hidden units, whose hidden and cell states are initialised using the vectors h^D and c^D. The first cell is fed with h^D as inputs. From the second cell on, the output of each LSTM cell is recursively used as input of the successive cell. The second stage of the decoder is a bidirectional LSTM layer of u_d hidden units, whose output sequences are fed to a final time-distributed dense layer, that brings the matrices to assume the output shape. The output of the network $\hat{X}_{(k+1:k+t^F)}$ is the prediction of $\bar{X}_{(k+1:k+t^F)}$, which represents the ground truth. The entries of $\hat{X}_{(k+1:k+t^F)}$ are denoted with $\hat{x}^j_i$, being it the value of the jth vehicle feature at the ith time step.

Finally, we define the two hyperparameters $u_{e,d}$ and ξ as

$$u_{e,d} = u_e + u_d, \tag{4}$$

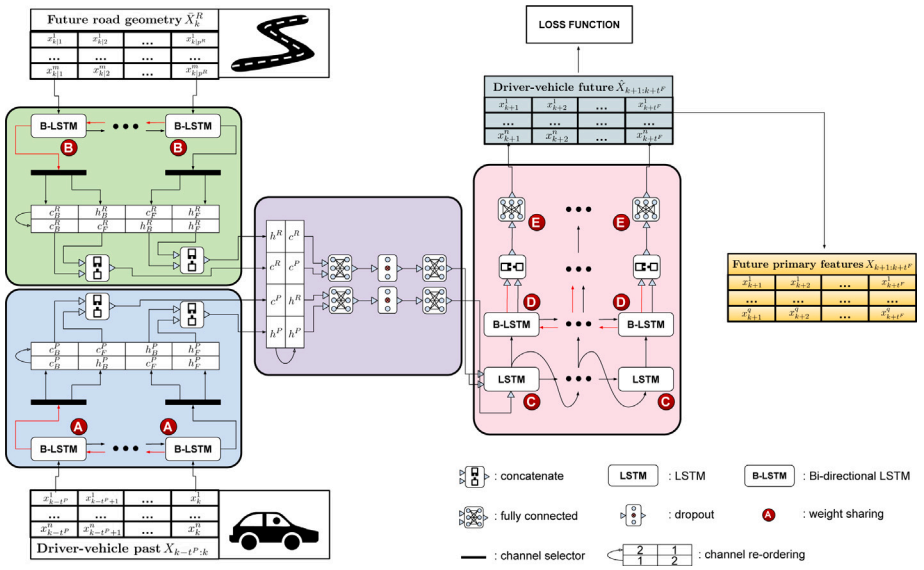


Fig. 2. Proposed deep neural network. The historical data (past) of the driver commands and vehicle states are encoded by a bi-directional LSTM to simultaneously represent the recent dynamics as well as the long-term context. In the same way, also the future road geometry is encoded by a bi-directional LSTM. The resulting cell states and hidden states of the two encoders are then passed through parallel fully connected layers, to fuse the past dynamics and future road information and generate the initial hidden and cell states for the decoder. The latter consists of an LSTM layer with recursion and a bi-directional LSTM. The output shape is finally obtained using fully connected layers.

$$\xi = \frac{u_e}{u_{e,d}}. \tag{5}$$

These two hyperparameters synthetically express the total network complexity for the encoding and decoding parts. Since the total network complexity influences the computational time needed to generate a prediction, it is important to properly limit the values of $u_{e,d}$ and ξ to render the proposed architecture feasible for online control applications. This consideration will be reminded in Section 5.2, where hyperparameters tuning is described.

4.2. Loss function

Taking advantage of how the problem is formulated in Section 3, the proposed prediction framework is potentially able to forecast the values of any vehicle feature that one could be interested in. Independently from the specific subset of q features of interest for a determined application, it is important to design a loss function that penalises prediction errors on the full set of n vehicle features. This choice forces the network to learn the driver-vehicle system dynamics, and infuses a natural generalisation into the data-driven approach. The q features of interest and the remaining $n-q$ ones are denominated as primary and secondary features, respectively.

To comply with the requirements of specific applications, in which one could be interested in maximising the prediction performance on the restricted set of q vehicle features, we employ Weighted Mean Square Error (WMSE) as loss function:

$$\mathcal{L} = \sum_{i=k+1}^{k+t^F} \frac{\sum_{j=1}^{n} w^j (\hat{x}_i^j - x_i^j)^2}{n}, \tag{6}$$

where $w^j = 1$ for $j = 1, \dots, q$ are the weights on the primary features, and $w^j = w$ for $j = q+1, \dots, n$ are the weights on the secondary features. The hyperparameter w is in the range $0 \leq w \leq 1$, and is tuned depending on the application. The employment of the weighting term w is strictly related to the requirements of online applicability of the prediction strategy. In case online applicability were not an issue, the network complexity could be arbitrarily increased to predict all of the n vehicle features. Instead, in the case study of this work we aim to design a network with prediction times that are suitable for online control applications; thus, we use w to help the network identify the relative priority between the primary and secondary features.

5. Experiments

The proposed framework is validated by applying it to a challenging case study, namely the prediction of the features that are commonly fed as references to any modern motion cueing algorithm for dynamic driving simulators based on Model Predictive

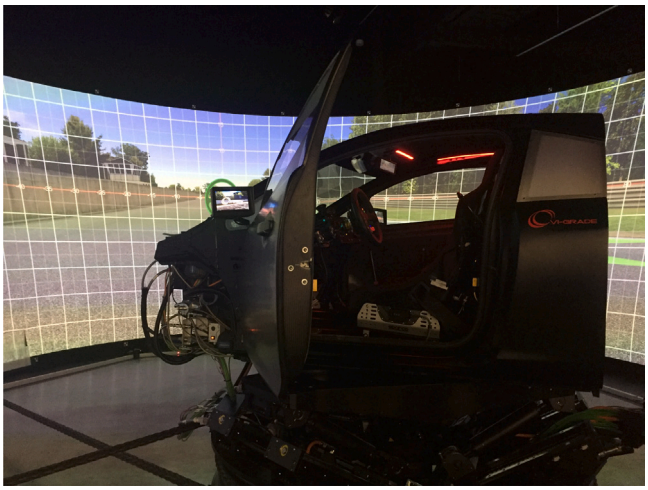


Fig. 3. Dynamic driving simulator used to collect the dataset.

Control (MPC). Our code and the dataset are online available.[1] We remark that the proposed methodology can also be employed to predict the evolution of any other feature that could be of interest for other automotive applications; we leave the study of further applications to future works.

As we remarked in Section 2, forecasting the vehicle accelerations and rotational velocities in driving simulators may significantly improve the simulation experience. For this reason, we will focus the attention of our network to prioritise the reconstruction of the vehicle CG longitudinal and lateral accelerations (a_X and a_Y, respectively), and vehicle chassis yaw rate (ω_y), being the in-plane motion the most crucial to enhance the simulator workspace utilisation. This means that the number of primary features is $q = 3$. We consider a prediction horizon length $t^F = 30$ (i.e. 3 s). According to the maximum vehicle speed, we estimate that a suitable choice for the hyperparameters that model the road geometry in the future is $d^R = 150$ m and $p^R = 50$.

5.1. Dataset

The dataset used to validate the prediction framework was generated on a dynamic driving simulator, shown in Fig. 3. The simulation environment set up for our analysis is the Canadian test track Calabogie, whose planimetry is shown in Fig. 4. We collected a dataset of 36 laps performed by a non-professional driver, for a total of approximately 75 minutes and 160 km travelled. A non-professional human driver was selected for the tests to introduce high variability into the resulting motion trajectories, which makes the prediction problem particularly challenging. The low level of repeatability of the driver at each lap is used to test the generalisation capability of the network. The latter is in fact expected to generate predictions that are close to the ground truth when the driving behaviour is more repeatable, and stabilise around an average behaviour when it is not.

The original data were acquired at a sampling frequency of 100 Hz. To be able to consider long-enough prediction horizons while saving the neural network computational time, data are downsampled at a frequency of 10 Hz, after using a zero-phase anti-aliasing filter with cut-off frequency at 4 Hz. As an example, Fig. 5 shows the original and downsampled frequency content of a_X, a_Y and ω_y. The downsampled signals contain approximately 98% of the power of the original ones; thus, the downsampling architecture allows to obtain a good trade-off between the level of detail in the representation of the driver-vehicle system dynamics and the speed of the prediction algorithm, in terms of online computational time at the test phase. Lastly, the signals are normalised to facilitate the neural network training process.

5.2. Training and validation of the model

The matrices defined in (1) are obtained by windowing each lap of the original dataset, using unitary shift and stride. A $31-4-1$ random split of the laps is applied to obtain the training, validation and test sets, respectively.

The model is trained using the Adam optimiser (Kingma and Ba, 2015), with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-7}$. After a first coarse tuning, it was observed that the hyperparameters affecting the network performance are the dropout rate r, the weight w on the secondary features, the number of time steps of the past t^P, and the hyperparameters $u_{e,d}$ and ξ defining the structure of the network. To fine tune the hyperparameters, we employ Bayesian optimisation (Pelikan and Goldberg, 1999) (using the toolbox (Nogueira,

[1] https://github.com/lpaparusso/DriVe-forecast

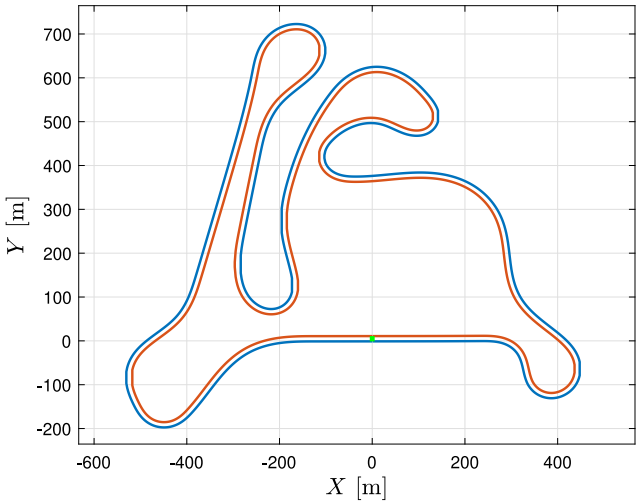


Fig. 4. Planimetry of the test track Calabogie used for the validation of the proposed approach. The right and left margins are shown in red and blue, respectively, i.e. the lap is clockwise. The start line is marked in green.

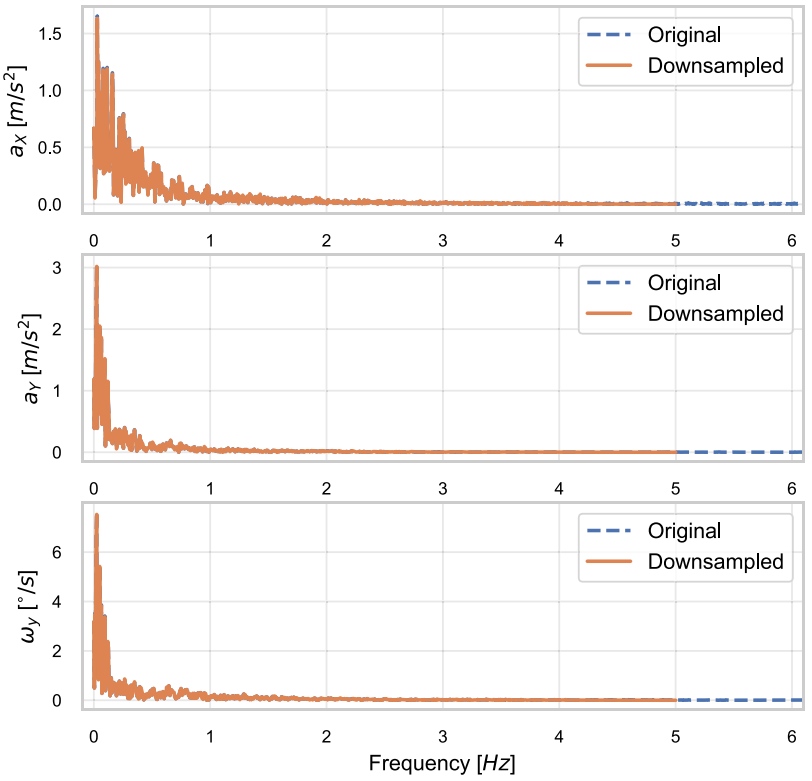


Fig. 5. Frequency content of the longitudinal acceleration a_X, lateral acceleration a_Y and yaw rate ω_y, for the original and downsampled signals. The downsampled signals contain approximately 98% of the power of the original ones.

2014)), which targets finding the best value of the metrics $\mathcal{M}$ on the validation set. The metrics $\mathcal{M}$ is the Mean Absolute Error (MAE) referred to the primary features only, i.e. a_X, a_Y and ω_y,

$$\mathcal{M} = \sum_{i=k+1}^{k+t^F} \frac{\sum_{j=1}^{q} |\hat{x}_i^j - x_i^j|}{q}, \tag{7}$$

being $|\cdot|$ the absolute value operator. Bayesian optimisation implements 2 steps of random exploration and 10 iterations.

Table 1
Range of the hyperparameters in Bayesian optimisation, and their optimised values in the original and ablated experiments.

Hyperparameter	min	max	noRoad	roadCoord	Ours
Dropout rate r	0.25	0.5	0.27	0.39	0.25
Secondary features weight w	0.0	1.0	0.40	0.54	**0.23**
Past horizon t^P	15	40	37	17	37
Network complexity $u_{e,d}$	70	110	94	92	97
Encoder relative complexity ξ	0.3	0.5	0.39	0.36	0.31

Table 2
Sample average and standard deviation of the metrics using cross-validation.

Model	$\mathcal{M}$	LONG-E [m/s^2]	LAT-E [m/s^2]	YAW-E [rad/s]	Off-Road Rate
noRoad	0.1668(±0.0079)	0.7693(±0.0656)	0.6956(±0.0958)	2.1403(±0.2655)	0.68(±0.04)
roadCoord	0.1670(±0.0076)	0.8865(±0.0307)	0.7269(±0.0859)	2.2612(±0.2261)	0.69(±0.03)
Ours	**0.1572(±0.0059)**	**0.7112(±0.0952)**	**0.6270(±0.0773)**	**2.0103(±0.2491)**	**0.53(±0.04)**

$\mathcal{M}$: MAE metrics on the normalised data; LONG-E: MAE on the longitudinal acceleration; LAT-E: MAE on the lateral acceleration; YAW-E: MAE on the yaw rate.

The training process of each model generated by Bayesian optimisation is performed in two phases. In the first phase, the Adam optimiser starts from a random initialisation of the weights of the neural network; to re-initialise the optimiser with weights that are closer to the optimal ones, which favours a more precise gradient descent, the second phase starts from the final configuration of the first phase. We now detail the characteristics of the two training phases. The first phase implements an exponentially decaying learning rate, going from 0.001 to 0.0005 in 34 epochs; after epoch 34, the learning rate is kept constant. We set the maximum number of epochs to 100, while using an early stopping option that activates when the validation metrics does not decrease within 25 consecutive epochs. The second phase employs a constant learning rate equal to 0.0001, with the same early stopping option as the first phase and a maximum number of epochs equal to 500. A batch size equal to 64 showed the best performance.

Considering the high variability of motion of the non-professional driver and the relatively small size of the dataset, we perform cross-validation to obtain performance indices that do not depend on the specific data split. Therefore, fixing the optimal hyperparameters obtained through Bayesian optimisation, training is finally repeated 10 times, changing the training, validation and test sets at each iteration. This allows to better analyse the framework performance and generate meaningful statistics. These values will be shown in the next sections.

5.3. Ablation experiment

An ablation experiment is carried out to certify the effectiveness of the neural network structure in generating context-aware predictions. We create two types of ablation experiment:

- removing the components of the network corresponding to the future road geometry. With reference to Figure 2, the green block is removed, and its outputs are treated as empty arrays. We call this experiment noRoad;
- using a road representation in local coordinates with respect to the ego-vehicle, instead of the proposed set of road features described in Sec. 3.1; we call this experiment roadCoord.

The range of variation of each hyperparameter in Bayesian optimisation, and their optimised values for the original and ablated experiments, are reported in Table 1. We highlight that the ranges of the hyperparameters $u_{e,d}$ and ξ are properly set to reduce the complexity of the neural network, thus allowing to use the prediction approach in online control applications.

Considering the ablation noRoad and our complete model, it is interesting to notice the alignment between the optimised hyperparameters. The optimal number of steps in the past is $t^P = 37$ in both cases. This highlights that it is sufficient to observe 3.7 s of the past to characterise the current driving condition/situation, which is likely for the dynamics of the driver-vehicle system. The only hyperparameter that differs in the two experiments is the secondary features weight w. It is reasonable that the noRoad experiment employs higher weights for the secondary features, as it cannot contextualise the dynamics according to the road. Thus, the noRoad neural network has to focus more intensively on the relationship between all of the variables, i.e. the vehicle dynamics, to support future predictions.

For the same reason, also the ablation experiment roadCoord uses a higher weight on the secondary features weight w. Moreover, the past horizon t^P reduces to 17 time steps.

We evaluate the models in 10 cross-validation iterations, and report the results in Table 2. As expected, the proposed model shows better performance on the validation set compared to the ablated models.

To compute the off-road rate, we infer the vehicle pose and orientation by integration of the predicted primary features. The corresponding differential equations are obtained by means of simple kinematic considerations. As shown in Table 2 and Fig. 6, the proposed road geometry encoder helps keeping the vehicle trajectories within the road margins. However, we must highlight that our off-rate performance is inferior to state-of-the-art pose predictors. This result was expected, since our model focuses on dynamics prediction and not on pose prediction.

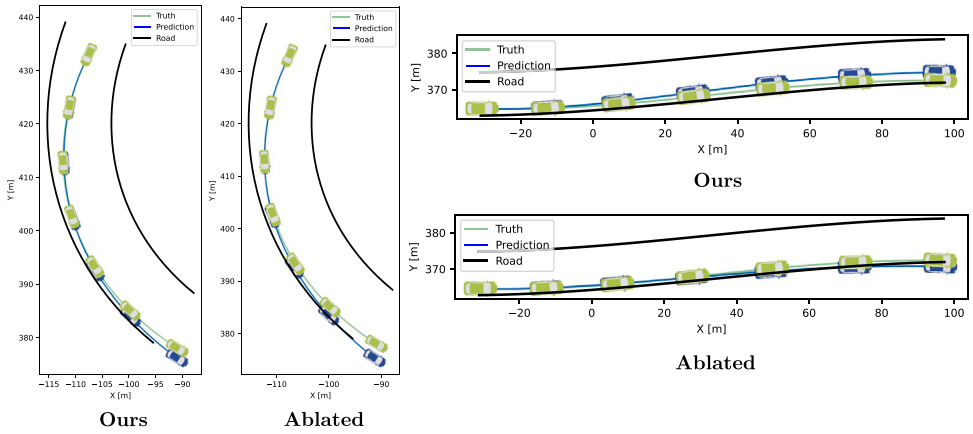


Fig. 6. Simulated vehicle trajectories obtained by integration of the predicted primary features. We compared Ours with noRoad. The complete model helps to better maintain the vehicle trajectories within the road margins.

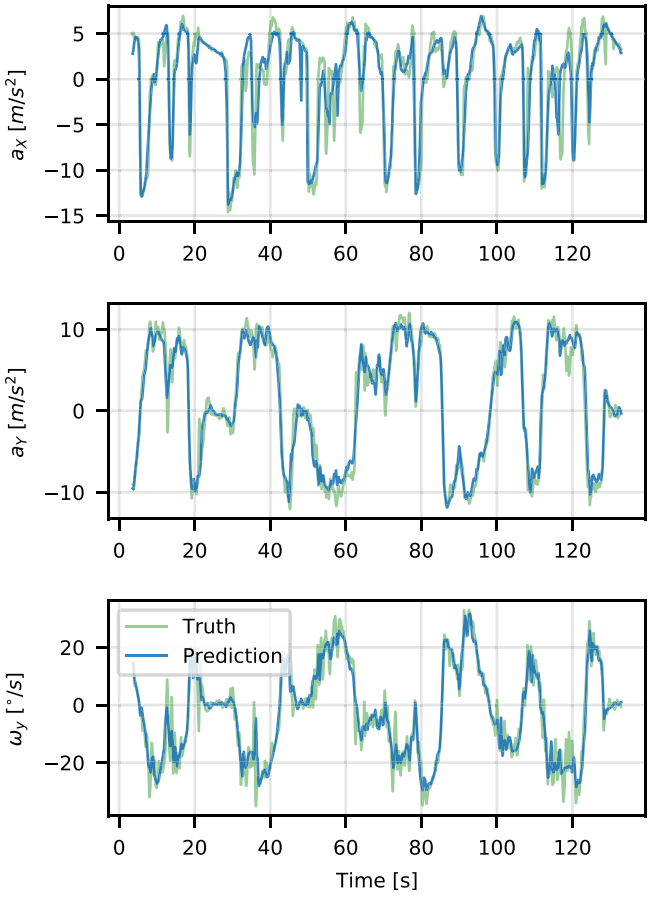


Fig. 7. Prediction of the primary features along the test lap. Each point of the blue curves represents the value predicted 30 time steps (3 s) in advance.

5.4. Analysis of the model performance

The prediction of the variables of interest, i.e. a_X, a_Y and ω_y, along the test lap is reported in Fig. 7 for the two prediction horizons. Despite the highly variable behaviour of the non-professional driver, the proposed prediction framework succeeds in distinguishing the different portions of the road and generates context-aware predictions. As expected, the forecasting performance is higher in the

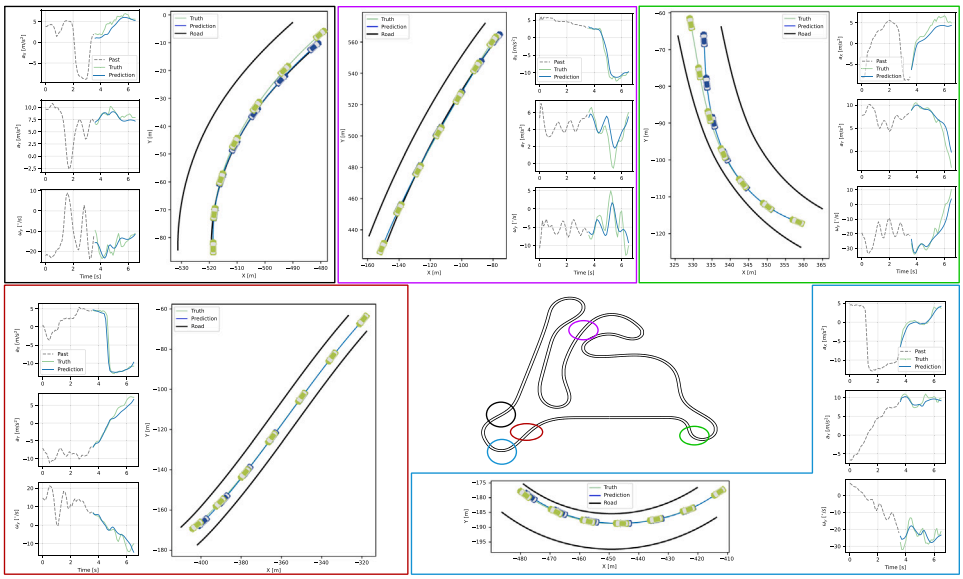


Fig. 8. Exemplary prediction horizons along the track extracted from the test set. The predictor shows good accuracy even at the latest time steps of the prediction horizons.

points of the track where the driver behaviour is more repeatable. Instead, the network filters out the fast oscillations that derive from sporadic actions, stabilising around an average value that still properly captures the dominant trend of the dynamics.

To better visualise the forecasting performance in whole prediction horizons, we report some examples in Fig. 8. Moreover, we report the boxplots (mean and standard deviation) of the prediction error along the future horizon in Figure 9. The prediction performance is shown to have a decay over the prediction horizon, which is common in open-loop prediction. Still, the performance decay is mild, which suggests that the framework is able to give importance to both short-term and long-term predictions.

To further motivate our work, and the fact that dynamics prediction is a different problem compared to pose prediction, we create another comparative example. We train a new model to predict pose only. Then, we derive the accelerations from the predicted poses. The results, reported in Figure 10, show that the lateral acceleration profile cannot be correctly reconstructed.

Finally, we evaluate the computational time of the neural network at the test phase. The results obtained for 40 consecutive windows are shown in Fig. 11. Considering the neural network trained for $t^F = 30$, the average computational time of a prediction window is 32 ms, with a sample standard deviation of 2 ms. The tests were run using interpreted Python code on a PC with an Intel Core i7-8565U CPU and 16 GB RAM. The measured computational times show that the proposed method is already suitable for simulation and online control applications. We remark that it may be possible to further increase performance by resorting to compiled code and executing the operations on dedicated GPUs.

We finally provide a baseline to highlight the reduction of computational time achieved by our method. We run a multibody simulation with the commercial software VI-Grade, considering the same prediction horizon (3 s) and known control inputs (straight acceleration). Simulating the 3-seconds future takes around 500 ms on the same PC. Although there might exist more efficient implementations, this example represents a fair baseline, as it employs a commercial software and the same computing device of our framework. We also remark that, if the control inputs are unknown, as in the problem investigated in this work, the computational time may become even higher, due to step-by-step computations. This highlights the benefits introduced by our approach, which forecasts the driver-vehicle dynamics at once for a multi-step future.

6. Generalisation on new drivers and tracks

Generalisation on the driver's behavioural variability, which is the core of this work, has already been investigated in the previous section. However, as an extra benefit of our approach, we show that the trained neural network can generalise with fair accuracy on new drivers and tracks, despite the limited amount of data seen at training time. In other words, the trained neural network with $t^F = 30$ is used to predict state trajectories from a new dataset, generated under considerably different high-level conditions with respect to the ones employed for training and validation. In particular, we evaluate the prediction performance of the network by changing driver and track, respectively, in two separated tests. This check is meant to assess how far the current training/validation set is from an ideal one whose range of variability allows to achieve high prediction performance on a generic high-level scenario.

The new track is obtained by travelling the original test track in opposite direction. For completeness, it has to be remarked that a track travelled in opposite direction generates driver-vehicle dynamics that are totally different and independent from the ones experienced in the original direction.

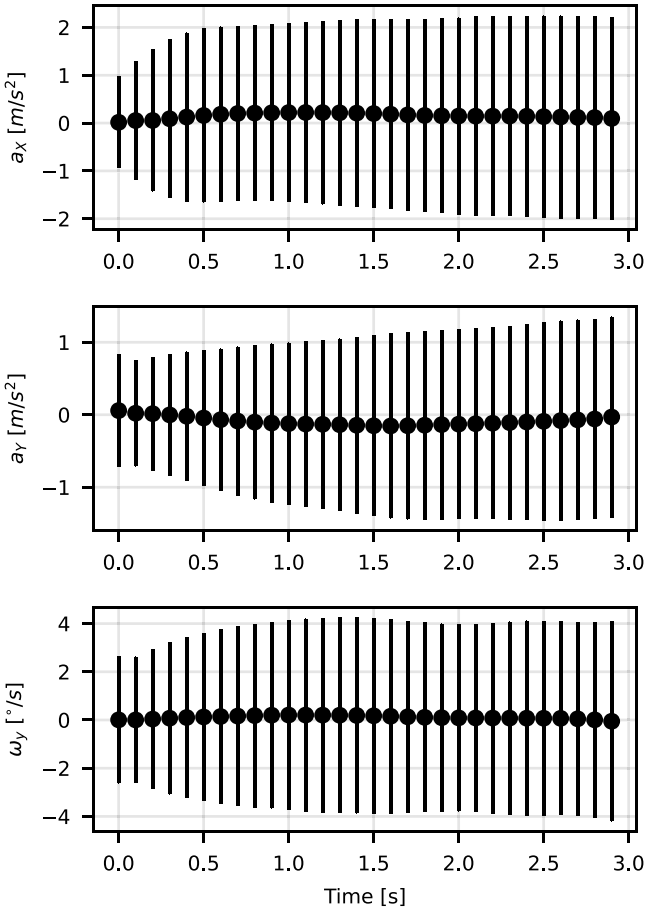


Fig. 9. Mean-Standard-Deviation boxplots of the prediction error of the primary features along the future horizon.

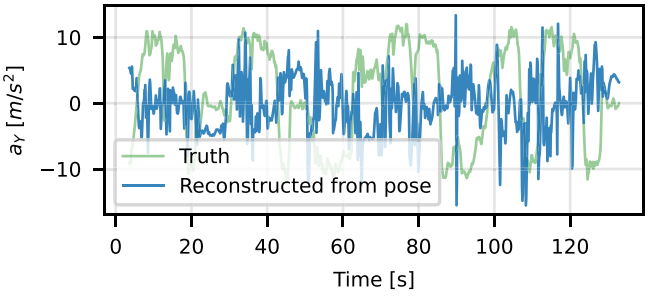


Fig. 10. Derivation of the lateral accelerations from pose prediction. Each point of the blue curves represents the value predicted 30 time steps (3 s) in advance. The example shows the importance of predicting dynamical quantities directly, instead of deriving them from pose forecasts.

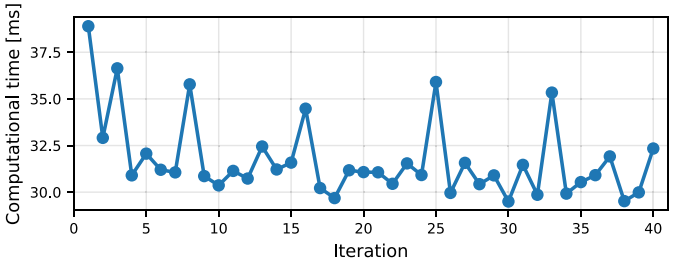
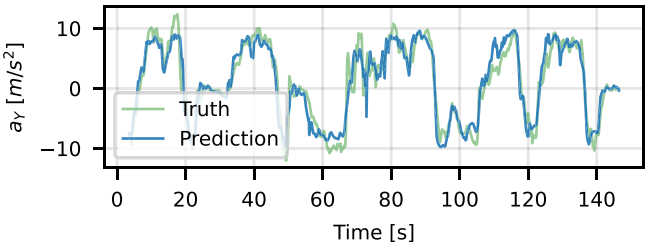
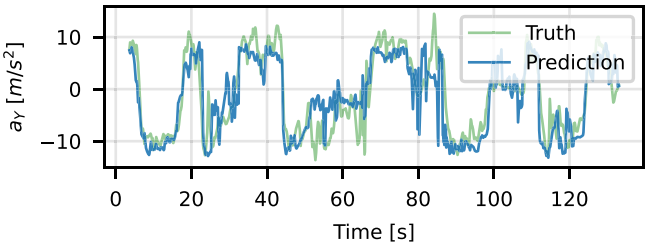


Fig. 11. Computational time of the neural network for 40 consecutive prediction horizons at test time.



(a)



(b)

Fig. 12. Prediction of the lateral acceleration, using a test lap performed by a different driver (a) and on a different track (b). Each point of the orange curves represents the value predicted 30 time steps earlier.

Despite the track is fixed, the driving style of the new driver is completely different. This is due to the fact that both drivers are not professional drivers, and therefore cannot fully optimise motion on the track. This results in different braking points, gear shifts and curve exit points, to name a few.

The prediction performance for the lateral acceleration a_Y is shown in Fig. 12 for the two cases. The network shows good generalisation on the two high-level variations. Particularly, it is interesting how our framework manages to reconstruct the main trend of the signal on the new track, despite the training set contained only data from a single track, which represent few of the possible combinations of road geometry. This confirms that the proposed architecture succeeds in linking the decisions of the driver with the geometry of the road. Thus, by selecting a suitable set of tracks to populate the dataset, it may be possible to optimise the generalisation capabilities of the network, making it robust to new tracks. This analysis is left to future works.

As an interesting future work, transfer learning (Tan et al., 2018) may be used to optimally adapt the pre-trained neural network to different high-level conditions. Also, clustering algorithms can be employed to distinguish driving styles and provide additional information to the neural network.

7. Conclusions

This paper presented a deep-learning framework to simulate and predict dynamics of the driver-vehicle coupled system on a 3D road at once. It is based on a flexible problem formulation, so that the user can choose a generic set of signals to be forecast. The prediction problem has been solved with a deep neural network, whose complexity is governed through Bayesian Optimisation to allow for use in real-time control applications. The computational times measured during tests have confirmed the suitability of the method.

The performance of the proposed approach has been evaluated on a case study centred on motion cueing algorithms, with the dataset being generated in test sessions of a non-professional human driver on a dynamic driving simulator. The scheme has shown good performance in predicting the desired signals, namely the longitudinal and lateral accelerations of the vehicle, and its yaw rate. An ablation study has also demonstrated the appropriateness of the neural network structure.

The generalisation capabilities of the framework on two new test sets, generated by changing driver and track, respectively, have been analysed. Despite the few cases seen by the neural network at training time, the framework has been capable of producing suitable forecasts on the new test sets.

Future works will be divided into two strands of activities. The first targets performance improvement. We shall define a minimal set of drivers and tracks to possibly improve the generalisation capabilities of the neural network on any new driver/track configuration. Also, the design of methods to enhance an automatic adaptation of the network to different drivers and tracks will be investigated. Moreover, the deep learning architecture will be extended to model the future state trajectories according to probabilistic reasoning.

The second activity strand targets framework extension to incorporate urban roads and traffic participants, which both condition the ego-vehicle's behaviour and dynamics in urban traffic settings. Both have been widely studied in literature, and incorporated into state-of-the-art prediction frameworks, by encoding their features through NN models. Similarly, since our framework is based on an encoder–decoder structure too, we plan to substitute our road encoder with state-of-the-art encoders of the traffic agents and the urban road.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Asadi, H., Mohamed, S., Lim, C.P., Nahavandi, S., 2017. Robust optimal motion cueing algorithm based on the linear quadratic regulator method and a genetic algorithm. IEEE Trans. Syst. Man Cybern.: Syst. 47 (2), 238–254. http://dx.doi.org/10.1109/TSMC.2016.2523906.

Bao, Z., Hossain, S., Lang, H., Lin, X., 2023. A review of high-definition map creation methods for autonomous driving. Eng. Appl. Artif. Intell. 122, 106125. http://dx.doi.org/10.1016/j.engappai.2023.106125.

Beghi, A., Bruschetta, M., Maran, F., 2012. A real time implementation of MPC based motion cueing strategy for driving simulators. In: 2012 IEEE 51st IEEE Conference on Decision and Control. CDC, pp. 6340–6345. http://dx.doi.org/10.1109/CDC.2012.6426119.

Beghi, A., Bruschetta, M., Maran, F., 2013. A real-time implementation of an MPC-Based motion cueing strategy with time-varying prediction. In: 2013 IEEE International Conference on Systems, Man, and Cybernetics. pp. 4149–4154. http://dx.doi.org/10.1109/SMC.2013.707.

Blundell, M., Harty, D., 2004. The Multibody Systems Approach to Vehicle Dynamics. Elsevier, Oxford.

Braghin, F., Cheli, F., Melzi, S., Sabbioni, E., 2008. Race driver model. Comput. Struct. 86 (13), 1503–1516. http://dx.doi.org/10.1016/j.compstruc.2007.04.028.

Britz, D., Goldie, A., Luong, M.-T., Le, Q., 2017. Massive exploration of neural machine translation architectures. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1442–1451. http://dx.doi.org/10.18653/v1/D17-1151.

Bruschetta, M., Cenedese, C., Beghi, A., 2019. A real-time, MPC-based Motion Cueing Algorithm with look-ahead and driver characterization. Transp. Res. F 61, 38–52. http://dx.doi.org/10.1016/j.trf.2017.04.023.

Cacciabue, C., 2007. Modelling Driver Behaviour in Automotive Environments: Critical Issues in Driver Interactions with Intelligent Transport Systems. Springer-Verlag, London.

Casas, S., Sadat, A., Urtasun, R., 2021. MP3: A unified model to map, perceive, predict and plan. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, pp. 14403–14412.

Dagdelen, M., Reymond, G., Kemeny, A., Bordier, M., Maïzi, N., 2009. Model-based predictive motion cueing strategy for vehicle driving simulators. Control Eng. Pract. 17 (9), 995–1003. http://dx.doi.org/10.1016/j.conengprac.2009.03.002.

Deo, N., Wolff, E., Beijbom, O., 2022. Multimodal trajectory prediction conditioned on lane-graph traversals. In: Faust, A., Hsu, D., Neumann, G. (Eds.), Proceedings of the 5th Conference on Robot Learning. In: Proceedings of Machine Learning Research, vol. 164, PMLR, pp. 203–212.

Dominguez-Sanchez, A., Cazorla, M., Orts-Escolano, S., 2017. Pedestrian movement direction recognition using convolutional neural networks. IEEE Trans. Intell. Transp. Syst. 18 (12), 3540–3548. http://dx.doi.org/10.1109/TITS.2017.2726140.

Gillespie, T.D., 1992. Fundamentals of Vehicle Dynamics. SAE International, Warrendale.

Grottoli, M., Cleij, D., Pretto, P., Lemmens, Y., Happee, R., Bülthoff, H., 2018. Objective evaluation of prediction strategies for optimization-based motion cueing. Simulation: Trans. Soc. Model. Simul. Int. 95, http://dx.doi.org/10.1177/0037549718815972.

Ivanovic, B., Elhafsi, A., Rosman, G., Gaidon, A., Pavone, M., 2020. MATS: An interpretable trajectory forecasting representation for planning and control. In: Conference on Robot Learning. CoRL.

Jiao, J., 2018. Machine learning assisted high-definition map creation. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference, Vol. 01. COMPSAC, pp. 367–373. http://dx.doi.org/10.1109/COMPSAC.2018.00058.

Kingma, D., Ba, J., 2015. Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations. ICLR.

Kutz, J., 2017. Deep learning in fluid dynamics. J. Fluid Mech. 814, 1–4. http://dx.doi.org/10.1017/jfm.2016.803.

Li, Q., Wang, Y., Wang, Y., Zhao, H., 2022. HDMapNet: An online HD map construction and evaluation framework. In: 2022 International Conference on Robotics and Automation. ICRA, pp. 4628–4634. http://dx.doi.org/10.1109/ICRA46639.2022.9812383.

Liu, R., Wang, J., Zhang, B., 2020. High definition map for automated driving: Overview and analysis. J. Navig. 73 (2), 324–341. http://dx.doi.org/10.1017/S0373463319000638, Publisher: Cambridge University Press.

Milliken, W.F., Milliken, D.L., 1994. Race Car Vehicle Dynamics. SAE International, Warrendale.

Mohammadi, A., Asadi, H., Mohamed, S., Nelson, K., Nahavandi, S., 2019. Multiobjective and interactive genetic algorithms for weight tuning of a model predictive control-based motion cueing algorithm. IEEE Trans. Cybern. 49 (9), 3471–3481. http://dx.doi.org/10.1109/TCYB.2018.2845661.

Morton, J., Wheeler, T.A., Kochenderfer, M.J., 2017. Analysis of recurrent neural networks for probabilistic modeling of driver behavior. IEEE Trans. Intell. Transp. Syst. 18 (5), 1289–1298. http://dx.doi.org/10.1109/TITS.2016.2603007.

Mozaffari, S., Al-Jarrah, O.Y., Dianati, M., Jennings, P., Mouzakitis, A., 2020. Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. IEEE Trans. Intell. Transp. Syst. 1–15. http://dx.doi.org/10.1109/TITS.2020.3012034, Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Nogueira, F., 2014. Bayesian Optimization: Open source constrained global optimization tool for Python. URL https://github.com/fmfn/BayesianOptimization.

Pelikan, M., Goldberg, D.E., 1999. BOA: The Bayesian optimization algorithm. In: GECCO'99: Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation, Vol. 1. pp. 525–532.

Plöchl, M., Edelmann, J., 2007. Driver models in automobile dynamics application. Veh. Syst. Dynam. 45 (7–8), 699–741. http://dx.doi.org/10.1080/00423110701432482.

Rudenko, A., Palmieri, L., Herman, M., Kitani, K.M., Gavrila, D.M., Arras, K.O., 2020. Human motion trajectory prediction: s survey. Int. J. Robot. Res. 39 (8), 895–935. http://dx.doi.org/10.1177/0278364920917446.

Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M., 2020. Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data. In: Computer Vision, Vol. 12363. ECCV 2020, Springer International Publishing, pp. 683–700. http://dx.doi.org/10.1007/978-3-030-58523-5_40, Series Title: Lecture Notes in Computer Science.

Scheel, O., Bergamini, L., Wolczyk, M., Osiński, B., Ondruska, P., 2021. Urban driver: Learning to drive from real-world demonstrations using policy gradients. In: 5th Annual Conference on Robot Learning.

Schwarting, W., Alonso-Mora, J., Paull, L., Karaman, S., Rus, D., 2018. Safe nonlinear trajectory generation for parallel autonomy with a dynamic vehicle model. IEEE Trans. Intell. Transp. Syst. 19 (9), 2994–3008. http://dx.doi.org/10.1109/TITS.2017.2771351, Conference Name: IEEE Transactions on Intelligent Transportation Systems.

Sun, J., Kim, J., 2021. Joint prediction of next location and travel time from urban vehicle trajectories using long short-term memory neural networks. Transp. Res. C 128, 103114. http://dx.doi.org/10.1016/j.trc.2021.103114.

Suo, S., Regalado, S., Casas, S., Urtasun, R., 2021. TrafficSim: Learning to simulate realistic multi-agent behaviors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, pp. 10400–10409.

Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C., 2018. A survey on deep transfer learning. In: Artificial Neural Networks and Machine Learning. ICANN 2018, In: Lecture Notes in Computer Science, vol. 11141, pp. 270–279. http://dx.doi.org/10.1007/978-3-030-01424-7_27.

Walker, J., Doersch, C., Gupta, A., Hebert, M., 2016. An uncertain future: Forecasting from static images using variational autoencoders. In: Computer Vision. ECCV 2016, In: Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 835–851. http://dx.doi.org/10.1007/978-3-319-46478-7_51.

Wang, P., Liu, H., Wang, L., Gao, R., 2018. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. CIRP Ann. - Manuf. Technol. 67, 17–20. http://dx.doi.org/10.1016/j.cirp.2018.04.066.

Xiao, Y., Codevilla, F., Gurram, A., Urfalioglu, O., López, A.M., 2022. Multimodal end-to-end autonomous driving. IEEE Trans. Intell. Transp. Syst. 23 (1), 537–547. http://dx.doi.org/10.1109/TITS.2020.3013234.

Yeung, E., Kundu, S., Hodas, N., 2019. Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In: 2019 American Control Conference. ACC, pp. 4832–4839. http://dx.doi.org/10.23919/ACC.2019.8815339.

Zhang, J., Liu, H., Chang, Q., Wang, L., Gao, R., 2020. Recurrent neural network for motion trajectory prediction in human-robot collaborative assembly. CIRP Ann. - Manuf. Technol. 69, 9–12. http://dx.doi.org/10.1016/j.cirp.2020.04.077.

Zyner, A., Worrall, S., Nebot, E., 2020. Naturalistic driver intention and path prediction using recurrent neural networks. IEEE Trans. Intell. Transp. Syst. 21 (4), http://dx.doi.org/10.1109/TITS.2019.2913166.