Systematic Model-based Design of a Reinforcement Learning-based Neural Adaptive Cruise Control System

Or Aviv Yarom[a], Jannis Fritz[1][b], Florian Lange and Xiaobo Liu-Henke
Ostfalia University of Applied Sciences, Salzdahlumer Str. 46/48, 38302 Wolfenbuettel, Germany

Keywords: Design Methodology, Artificial Neural Networks, Automated Driving, ACC, Automated Longitudinal Guidance, Artificial Intelligence, Neural Architecture Search, NAS.

Abstract: In this paper, the systematic model-based design of a reinforcement learning-based neuronal adaptive cruise control is described. Starting with an introduction and a summary of current fundamentals, design methods for intelligent driving functions are presented. The focus is on the first-time presentation of a novel design methodology for artificial neural networks in control engineering. This methodology is then applied and fully validated using the example of an adaptive cruise control system.

1 INTRODUCTION

The innovation alliance autoMoVe (Dynamically Configurable Vehicle Concepts for Use-Specific Autonomous Driving) (Raulf et. al., 2020), funded by the European Regional Development Fund (ERDF) (European Commission, 2014), aims to develop an autonomous, modular and electric vehicle concept. By exchanging application-specific modules at runtime, a wide range of applications from internal freight transport to passenger transport in road traffic is to be realized autonomously. Within the framework of this research project, the Ostfalia autoEVM sub-project (Holistic Electronic Vehicle Management for Autonomous Electric Vehicles) focuses on the model-based development of innovative intelligent algorithms and functions for autonomous driving.

With a higher degree of automation in driving, the requirements for the vehicle or the automated driving functions also increase. Current functions and algorithms based on methods of control theory or classical information processing can no longer fully meet these requirements (Milz and Schrepfer, 2020). Artificial intelligence (AI) therefore represents a key technology for many domains involved in the development and usage of intelligent, automated vehicles (Fayjie et. al., 2018).

Modern vehicles and driving functions are, regardless of the type of information processing, complex mechatronic systems with a high degree of internal and external networking. To master this complexity in the development and validation process, a systematic design approach is essential. On one hand, a verification-oriented and simulation-based design methodology established in mechatronics research is used. (Liu-Henke et. al., 2021) AI algorithms, especially artificial neural networks (ANN) used in this work, are very different from control theory approaches in their functionality and design process. Therefore on the other hand, a novel design methodology for ANNs is applied and presented for the first time in this paper.

As a sub-goal of the above mentioned research project, the present work applies the mentioned design methodologies for ANNs and mechatronic systems on a function for automated longitudinal guidance in terms of intelligent adaptive cruise control. The term intelligent refers on the one hand to the use of ANNs, which are modeled according to the functioning of the human brain, and on the other hand to the self-learning or experience-based training approach of reinforcement learning.

[a] https://orcid.org/0000-0001-5627-4199
[b] https://orcid.org/0000-0001-5613-4129

Yarom, O., Fritz, J., Lange, F. and Liu-Henke, X.
Systematic Model-based Design of a Reinforcement Learning-based Neural Adaptive Cruise Control System.
DOI: 10.5220/0010923300003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 3*, pages 889-896
ISBN: 978-989-758-547-0; ISSN: 2184-433X

2 STATE OF THE ART

2.1 Intelligent Driving Functions for Automated Longitudinal Guidance

With an increasing number and interconnection of advanced driver assistance systems (ADAS), the human driver successively delegates driving tasks to the vehicle until he finally becomes a passenger in autonomous driving. At that point, one no longer speaks of ADAS, but of (automated) driving functions. One example of such an automated driving function is adaptive cruise control (ACC). In contrast to simple cruise control, in which only a comparison between the set speed of the driver and the actual speed of the vehicle takes place, ACC can overwrite the set speed downwards by evaluating environmental sensors. The vehicle then maintains a speed-dependent safe distance to vehicles and objects in front by specifying reference values to subordinate systems of the longitudinal dynamics (drive, brake, transmission). (Lin, Nguyen and Wang, 2017)

Radar sensors are typically used for detecting obstacles ahead and the subsequent determination of relative speed and distance (Abdullahi and Akkaya, 2020). Modern systems often additionally use camera and lidar sensors for object detection. Another approach is the use of wireless vehicle-to-everything (V2X) communication to realize cooperative driving operation (Anayor, Gao and Odekunle, 2018).

From a control engineering point of view, ACC is a cascaded system in which the inner cascade controls a speed given by the outer cascade to regulate the safety distance. In practice, however, some driving functions exist that switch between distance keeping and pure speed control. But for highly automated driving, higher-order and holistic functions are advantageous. (Abdullahi and Akkaya, 2020) In the current literature, many approaches, such as a classical PI (Kiencke and Nielsen, 2005), non-linear model predictive (Shakouri and Ordys, 2014), or even neuro-fuzzy control (Lin, Nguyen and Wang, 2017) can be found for an ACC, each with specific advantages and disadvantages. In this paper, a novel holistic approach for an ACC using ANNs and reinforcement leaning will be investigated.

2.2 Basics of Artificial Neural Networks and Machine Learning

The term AI covers a large number of different methods and algorithms that deal with the autonomous and automated solving of problems (Fayjie et. al., 2018). ANNs and ML form a subfield of AI that has proven to be suitable for numerous problems in a wide variety of domains, including autonomous driving. Therefore, this paper focuses on this subfield. The numerous positive properties of ANNs and ML, such as adaptability, error resistance, versatility and above all learning ability, can be traced back to their similarity to the structure and functioning of the human brain.

Analogous to biology, (artificial) neurons are processing units that accumulate input stimuli via weighted connections and compute an output using an activation function. The interconnection of several neurons in at least two layers creates the ANN. Combinations of up to several hundred neurons in up to more than one hundred layers are common. Not only arbitrary forward but also time-feedback connections are possible in the ANN. The optimal architecture of an ANN cannot be determined analytically so far (Tirumala, 2020). Therefore, besides experience and test series, a systematic design methodology is necessary to find a suitable architecture in the trade-off between computational effort and performance capability. The number, interconnection and weighting of the connections characterizes the "intelligence" of an ANN. Generally speaking, more neurons and connections mean a higher performance of the ANN, while at the same time the computational effort increases.

Just like a human brain, the ANN must first learn or train a task. These terms refer to the adaptation of the connection weights. In the environment of autonomous driving, supervised learning (SL) and reinforcement learning (RL) are relevant for this. In SL, the ANN is presented with input data and the associated output. The ANN iteratively learns the relationship between these two quantities (Duriez, Brunton and Noack, 2017). This learning procedure is particularly suitable for image-based object recognition, for example (Lyu et. al., 2019). In RL, the ANN successively learns the optimal strategy from the experience of past sequences in terms of a given reward function (Duriez, Brunton and Noack, 2017). This method is used when no training data is available, e.g. in automated vehicle guidance (Huang et. al., 2019). SL and RL are upper categories of learning methods, with diverse training algorithms. Just like the ANN architecture, the optimal training algorithms as well as their parameters cannot be determined analytically. Thus, experience and experimentation are required here as well.

3 DESING METHODOLOGY

3.1 Model-based Controller Design

The complexity of modern vehicles is constantly increasing due to the higher degree of internal and external interconnection and the growing number of intelligent and powerful hardware and software components. In order to master the system complexity and to avoid errors at an early stage in the design of information processing, a holistic design methodology is indispensable. Therefore, the holistic, verification-oriented, model-based design methodology based on Rapid Control Prototyping (RCP) and Model-in-the-Loop (MiL), Software-in-the-Loop (SiL) and Hardware-in-the-Loop (HiL) simulations has become established. (Liu-Henke et. al., 2016)

The methodology builds on function-oriented physical models of a controlled system. The control function is then simulatively designed depending on the system behavior and validated in MiL simulations at an early stage. To avoid manual programming, the model and control function are developed in block diagram-based programming languages (Jacobitz and Liu-Henke, 2020). The subsequently automatically generated function code is tested again against the plant model in SiL simulations. HiL simulations are used for further validation and optimization of the information processing with real-time capable simulation models and real subcomponents of the system to be controlled (Liu-Henke et. al., 2016). The verification-oriented and iterative approach of this methodology also supports the development process in the challenging task of validation. The methodology addresses the weaknesses of classical validation based on physical prototypes, such as high resource requirements or safety risks for humans, machines and the environment. Due to their virtual character, MiL, SiL and HiL simulations save time and costs (Yarom et. al., 2020). They enable feasible and reproducible tests at any time without direct dependency on physical prototypes, day times or human experts. Thus, simulation runs can be automated for different functional variants or scenarios. This makes this methodology particularly suitable for training ANNs. This is because, with a few exceptions, machine learning is always iterative.

Virtual design methods like this form the basis for many intelligent systems, such as highly automated vehicles. With prototype-based testing, the hundreds of thousands of test miles required cannot be accomplished with reasonable time and cost. It should be noted that despite the use of this methodology, prototype-based tests cannot be completely eliminated, for example for certification tests. However, the number and the effort as well as the associated disadvantages can be reduced to a minimum. (Yarom et. al., 2020)

3.2 Systematic Design of ANNs

The design methodology for model-based controller design presented in the previous section 2.1 is a superordinate methodology. The design of an ANN can be considered as a part of this methodology, analogous to a controller design with dynamic compensation. ANN design is a complex process consisting of many design decisions, which individually or in combination have a major impact on the design outcome. These decisions are distributed over the entire development process of the ANN and concern e.g. the system structure (end-2-end function / combination of sub-functions), the learning method (SL / RL), the network architecture, the parameterization of the training algorithm or the evaluation of the ANN.

The special characteristic lies in the fact that in contrast to the very systematic-analytical design procedures of classical and modern control engineering, the decisions and their effects in ANN design are usually not mathematically-logically understandable for the human mind. As a result, there is no systematic design procedure for ANNs (Tirumala, 2020). Although one can find many applications of ANNs in current literature, information on the respective design processes is rare. When information is provided, the authors often follow a result-oriented empirical approach based on experience with low systematics.

The procedure model shown in Figure 1 presents a first approach to systematize the design process of ANNs. The starting point is the control problem specification. It hardly differs from the classical control theory. For example, especially in the case of RL, the work is typically model-based. The special feature is the choice of the system structure and interfaces. While these are often determined by physical conditions in classical controllers, they can theoretically be freely selected for ANNs. Subsequently, a suitable learning approach or a concrete learning algorithm must be chosen for the respective problem. There are many learning algorithms, with specific advantages and disadvantages, which have to be selected for each application. In RL, a basic distinction is made between gradient-based and gradient-free algorithms.

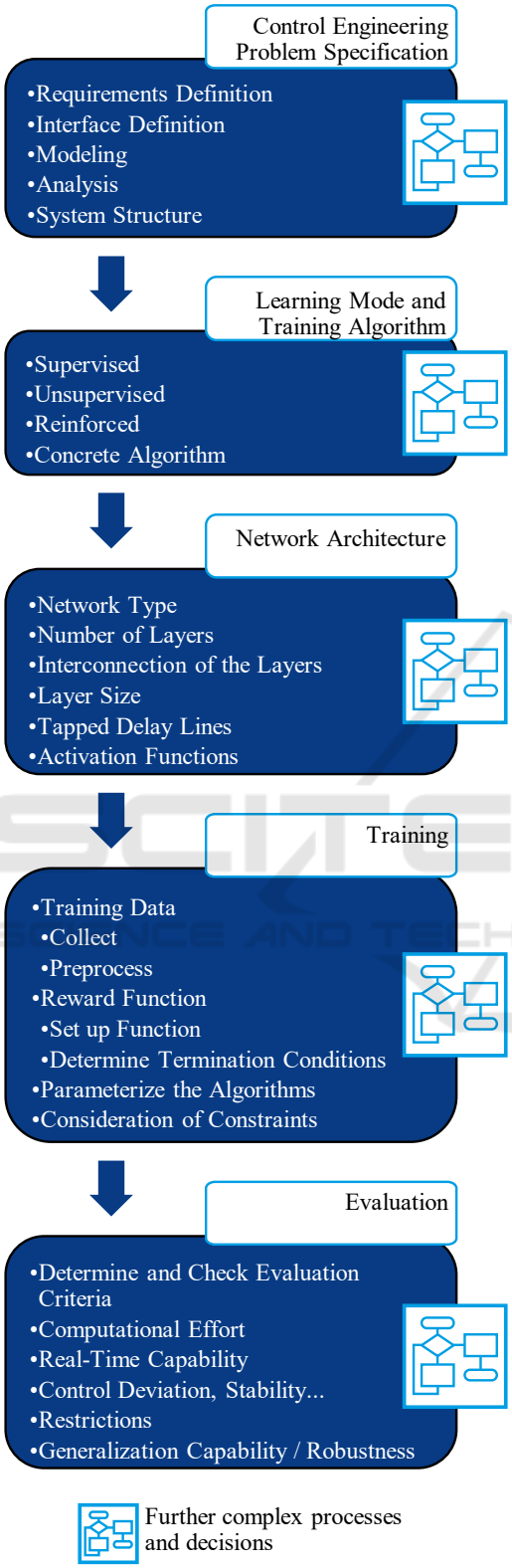

Figure 1: Procedure model for the systematic design of ANNs

The selection of the network architecture is the most complex part of the ANN design and has become a modern field of research in its own right. It is attempted to formulate the architecture selection as a search problem and to solve it automatically by means of a higher-level optimization technique. This is called Neural Architecture Search (NAS). (Rock et. al., 2021) After the network architecture has been determined, one can continue with the training, which is the actual core of the design. In this process, the behavior of the ANN is optimized with respect to the original purpose. In this step, especially the hyperparameters of the training algorithms have to be set. The last step is the evaluation of the ANN with respect to the original requirements and considering its application purpose, e.g. on a real-time system.

As already indicated for the NAS, each block in the process model from Figure 1 forms a higher-level process step and each contains further complex processes and decisions. Since this paper focuses on the application of an ANN as an ACC, only some of these will be taken up in more concrete terms later in this paper.

4 SYSTEMATIC MODEL-BASED DESIGN OF NEURAL ADAPTIVE CRUISE CONTROL

The aim of this paper is the systematic model-based design of a reinforcement learning-based neural adaptive cruise control system. This process is now presented according to the design methodology presented in Figure 1.

4.1 Control Engineering Problem Specification

4.1.1 Requirements Definition

The automated longitudinal guidance function presented and developed here is intended to meet the following requirements:

- Maintain a speed set by the driver and in no case exceed it
- Maintain a 2-second safety distance to a vehicle that may be driving ahead
- Do not fall below the safety distance, or only slightly and for a short period of time
- Follow any speed profile, even when maintaining the safety distance
- Maintain speed and distance without oscillations if possible

- The ego vehicle should realize accelerations in the range -3 m·s-2 to 2 m/s2
- Be based on ANNs and RL

4.1.2 System Structure and Interface Definition

Figure 2 schematically shows the system structure of the neural adaptive cruise control designed here, including the system environment. The starting point is a lead car, which moves with an arbitrary velocity v_{lead} and thus changes its position x_{lead} The actual center of the structure, however, is the ego vehicle model, which calculates the velocity v_{ego} and position x_{ego} of the ego vehicle based on acceleration forces F_A and braking forces F_B. These positions and velocities of the ego and lead car, are used by the radar sensor with "built-in" preprocessing to determine, within its range of 150 m, the input variables for the ANN. These are:

- The velocity error verr, i.e. the difference between the velocity vset specified by the driver and the actual ego velocity
- The integral of the speed error over time ∫ verr
- The ego speed v_{ego}

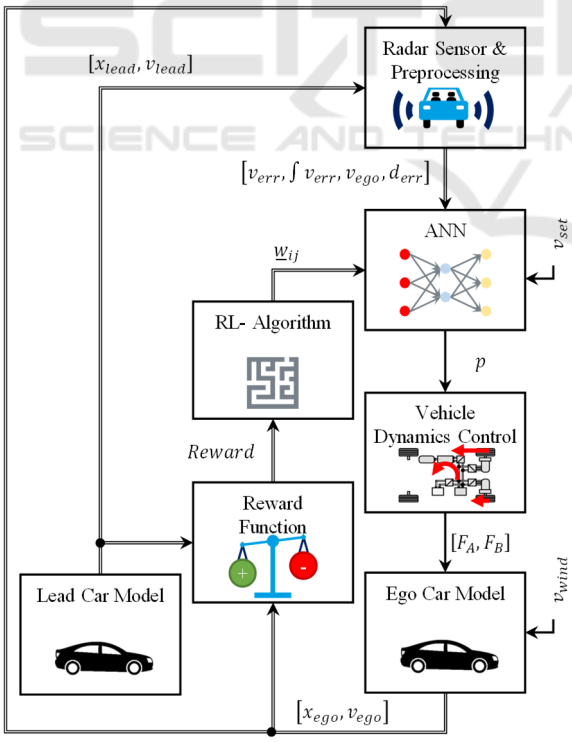

Figure 2: Functional structure of the reinforcement learning-based neural adaptive cruise control system.

- The distance error d_{err}, i.e., the difference between the safe distance d_{safe} and the actual distance between the vehicles d.

Using these input variables, the ANN then calculates a command value for the driving and brake pedal position $p \in \{-1 \leq p \leq 1 \mid \mathbb{R}\}$, which is converted into the above-mentioned longitudinal forces by a subordinate vehicle dynamics control system.

These are all components that the system uses in the application phase, i.e. after training. During training, two additional components are needed. The reward function, specified or to be worked out by the developer, which calculates a *reward* based on data of the system components in order to evaluate the ANN during training. The RL algorithm is an optimization technique that adjusts the weights $\underline{w}_{i,j}$ of the ANN to maximize the expected *reward* in the upcoming episode (one simulation run).

4.1.3 Modeling

The overall system shown in Figure 2 is modeled, simulated and trained in the simulation environment for the automated model configuration for the design and validation of AI-based driving functions from (Yarom and Liu-Henke, 2021). A classical longitudinal dynamics model is used to represent the behavior of the Ego and Lead vehicles:

$$m \cdot \ddot{x} = F_A - \underbrace{c_w \cdot A \cdot \frac{\rho}{2} \cdot \dot{x}^2}_{Air\ res.} - \underbrace{m \cdot \sin(\alpha)}_{Slope\ res.} - \underbrace{m \cdot \cos(\alpha) \cdot f_R}_{Rolling\ res.} - F_B. \quad (1)$$

The variable for the driving and brake pedal position p is scaled to the corresponding forces F_A and F_B. In order to be able to simulate a standstill and reversing, further mechanisms were implemented in the simulation environment, which will not be discussed in this paper.

The radar sensor with preprocessing is activated as soon as the distance between the vehicles falls below the range of the sensor. Here, on the one hand, situation-dependent reference values for distance and speed are calculated. For example, min(v_{set},v_{ego}) applies to the desired speed of the ego car if the ego car is in the range of d_{safe} to the lead car or below. On the other hand, the calculated set points are also compared with the actual values and deviations are calculated for the distance and the ego speed. Thus, this function component not only provides the input variables for the ANN, but also basic variables for the reward function.

4.1.4 Learning Mode and Training Algorithm

In principle, only RL algorithms can be used for this application. In order to create an optimal ACC function on the one hand and to gain further experience for the systematic design of ANNs on the other hand, different training algorithms will be used and compared. In preliminary tests with lower requirements the following training algorithms were used:

- Genetic Algorithm (GA)
- Particle Swarm Optimization (PSO)
- Deep Deterministic Policy Gradients (DDPG).

The first two methods are gradient-free, evolution-based algorithms. This class of algorithms usually evolves steadily toward a higher *reward*, but does not always converge to a true optimum. Their advantage is that you usually get to a good result fairly quickly. A special feature of the GA is that it is quite random and tends to find the global optimum rather than other algorithms if the parameters are set correctly. The DDPG, on the other hand, is a gradient-based algorithm that converges fairly reliably to a local optimum. However, it usually takes more time to do so. Moreover, it rarely reaches the global optimum.

The series of experiments showed that the DPPG achieves the best results with high probability. To make the results comparable, the same rudimentary reward function was used for all algorithms. While this series of experiments was already used to optimize the hyperparameters of the algorithms, the reward function was designed in detail at a later point.

4.2 Network Architecture

By choosing DDPG as training algorithm, two ANN architectures have to be chosen. This is because an actor and a critic network are required here. The actor network is the ANN that performs the actual control task. The critic network serves as a translator between the reward function and the algorithm for updating the connection weights. It is only needed during the training phase.

The critic network can usually be chosen as a simple feedforward network with rectifier functions as activation. Therefore, such an ANN with five layers was specified here as the critic network. The input variables are its output in addition to those of the actor network. The architecture of the actor network was automatically optimized using a NAS based on a GA. The result is a feed forward with four layers. In each of the first three layers there are 48 hidden neurons as well as rectifier functions for activation. In the last layer there is only one neuron with a *tanh* activation.

4.3 Training

Now that the network architectures have been determined, the reward function that determines the final behavior of the ANN or Ego Car must be designed. This consists of four terms:

- $-0.1 \cdot (v_{err})^2$: Penalizing speed errors to maintain the target speed.
- -(a)2: Penalization of accelerations, so that oscillations and abrupt velocity changes are avoided
- -0.1·(derr)2: Penalization of distance errors so that the target distance is maintained.
- +verr for verr ≤ 0.25 m/s: Reward of small speed errors so that the target speed is maintained.

In an iterative process, the ego car now runs through several episodes in which the actor network is used again and again as a longitudinal dynamics controller and receives a corresponding *reward* for its behavior. Figure 3 shows the course of the *rewards* over the episodes as well as the average reward. The *reward* is very negative at the beginning of the training, since the ANN does not yet behave well without prior knowledge. The *reward* quickly improves to the range of about -600 and then slowly increases until the ANN reaches about -91 reward points in the best episode at the end. In between there are occasional dips, which indicate a wrong weight change, but can be compensated by the algorithm. The fact that the best *reward* is -91 does not mean that the behavior of the ANN is bad. This is only caused by the design of the reward function. The course of the *reward* is therefore more crucial than its absolute value. Therefore, the evaluation after the training is essential.

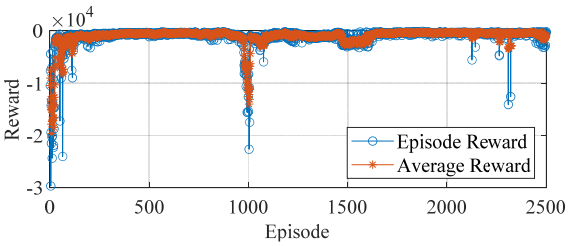

Figure 3: Course of the *reward* during training.

4.4 Evaluation

The evaluation is the final step in the design of the ANN for neural adaptive cruise control. A special aspect that must be considered in the evaluation is the so-called generalization capability. This means that the ANN should apply the learned knowledge to new, previously unknown problems. In order to be able to rate the generalization capability, the numerous necessary test scenarios for evaluation must therefore differ from the training situation.

During training, v_{set} was at 100 km/h and the lead car was driving ahead with an acceleration sine wave. In the one exemplary test situation shown here, v_{set} is increased to 108 km/h and the vehicle dynamics model of the Lead Car is replaced by a "hard" speed profile to further complicate the situation for the ANN. The speed profile of the Lead Car is shown in red in the upper part of Figure 4, v_{set} in blue. The yellow curve shows the speed of the Ego Car. In the lower part of the figure, the distances d_{safe} and d are plotted versus time. At the beginning, the Ego Car was deliberately placed very close ($d < d_{safe}$) behind the Lead Car, so that it must first maintain its speed to keep the safe distance. Subsequently, the Ego Car follows the speed curve of the Lead Car very well and stably until the Lead Car accelerates to 120 km/h, i.e., above v_{set}, at about 24 s. As desired, the ego car does not exceed v_{set} and the distance increases. Then, the Lead Car brakes abruptly and the Ego Car also brakes hard to maintain the safety distance without violating the acceleration requirement (section 4.1). Finally, the lead car accelerates again. The Ego Car controls v_{set} in a stable manner.

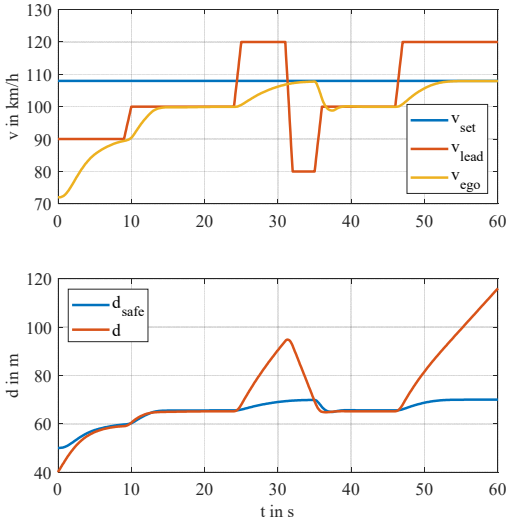

Figure 4: Simulation result of the neural adaptive cruise control system.

This means that the ANN has learned the adaptive cruise control and satisfies all requirements as expected. Further simulation results, which could not be presented here, have shown that the designed function is generalization capable. Thus, the novel methodology for systematic design of ANNs (in the first phase) can also be considered validated.

5 CONCLUSION AND OUTLOOK

In this paper, the systematic model-based design of a reinforcement learning-based neural adaptive cruise control system was presented. Starting with an introduction, the state of the art and the underlying design methodology were summarized. This relates to model-based controller design and systematic design of ANNs. The latter was presented as a novelty for the first time in this paper.

Subsequently, the application and validation of this methodology was carried out on the example of a neural adaptive cruise control system. The design process was described in detail and the resulting function was intensively evaluated and validated against the requirements.

Future work steps include the further validation of the design methodology as well as the design of further automated driving functions.

ACKNOWLEDGEMENTS

This publication resulted from the subproject "autoEMV" (Holistic Electronic Vehicle Management for Autonomous Electric Vehicles) in the context of the research project "autoMoVe" (Dynamically Configurable Vehicle Concepts for a Use-specific Autonomous Driving) funded by the European Fund for Regional Development (EFRE | ZW 6-85030889) and managed by the project management agency Nbank.

REFERENCES

Abdullahi, A., Akkaya, S. (2020). Adaptive Cruise Control: A Model Reference Adaptive Control Approach. *2020*

24th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania.

Anayor, C., Gao, W., Odekunle, A. (2018). Cooperative Adaptive Cruise Control of A Mixture of Human-driven and Autonomous Vehicles. *SoutheastCon 2018*, St. Petersburg, FL, USA.

Duriez, T., Brunton, S., Noack, B. R. (2017). Machine Learning Control. Springer International Publishing, Cham, Switzerland.

European Commission (2014). European Regional Development Fund. Available at: https://ec.europa.eu/regional_policy/en/funding/erdf/ (Accessed: 22 December 2021).

Fayjie, A. R., Hossain, S., Oualid D., Lee, D. (2018). Driverless Car: Autonomous Driving Using Deep Reinforcement Learning in Urban Environment. *2018 15th International Conference on Ubiquitous Robots (UR)*, Honolulu, Hawaii.

Huang, Z., Xu, X., He, H., Tan, J., Sun, Z. (2019). Parameterized batch reinforcement learning for longitudinal control of autonomous land vehicles. In *IEEE Trans. Syst., Man, Cybern, Syst.*, vol. 49, no. 4.

Jacobitz, S., Liu-Henke, X. (2020). The seamless low-cost development platform LoRra for model based systems engineering, *8th International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*, Valletta, Malta.

Kiencke, U., Nielsen, L. (2005). Automotive control systems: for engine driveline and vehicle. Springer, Berlin, 2005.

Lin, Y.-C., Nguyen, H.-L. T., Wang, C.-H. (2017). Adaptive neuro-fuzzy predictive control for design of adaptive cruise control system. *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*, Calabira, Italy.

Liu-Henke, X., Scherler, S., Fritsch, M., Quantmeyer, F. (2016). Holistic development of a full active electric vehicle by means of a model-based systems engineering. *2016 IEEE International Symposium on Systems Engineering (ISSE)*, Edinburgh, UK.

Liu-Henke, X., Jacobitz, S., Scherler, S., Göllner, M., Yarom, O., Zhang, J. (2021). A Holistic Methodology for Model-Based Design of Mechatronic Systems in Digitized and Connected System Environments. *16th International Conference on Software Technologies (ICSOFT)*, online.

Lyu, H., Fu, H., Hu, X., Liu, L. (2019). Edge-Based Segmentation Network for Real-Time Semantic Segmentation in Traffic Scenes. *2019 IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan.

Milz, S., Schrepfer, J. (2020). Is artificial intelligence the solution to all our problems? Exploring the applications of AI for automated driving. In Bertram T. (eds) *Automatisiertes Fahren 2019*. Springer Vieweg, Wiesbaden, Germany.

Raulf, C., Pethe, C., Vietor, T., Henze, R. (2020). Dynamically Configurable Vehicle Concepts for Autonomous Driving. *ATZ Worldwide,* vol. 122.

Rock, J., Roth, W., Toth, M., Meissner, P., Pernkopf, F. (2021). Resource-Efficient Deep Neural Networks for Automotive Radar Interference Mitigation. *in IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 4.

Shakouri, P., Ordys, A. (2014). Nonlinear model predictive control approach in design of adaptive cruise control with auto-mated switching to cruise control. *Control Engineering Practice*, vol. 26.

Tirumala, S.S. (2020). Evolving deep neural networks using co-evolutionary algorithms with multi-population strategy. In *Neural Comput & Applic, vol. 32*.

Yarom, O. A., Scherler, S., Goellner, M., Liu-Henke, X. (2020). Artificial Neural Networks and Reinforcement Learning for model-based design of an automated vehicle guidance system. *12th International Conference on Agents and Artificial Intelligence (ICAART)*, Valletta, Malta.

Yarom, O., Liu-Henke, X. (2021). *Development of a Simulation Environment for Automated Model Configuration for the Design and Validation of AI-Based Driving Functions. 11th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH)*, online.