*EVS30 Symposium*
*Stuttgart, Germany, October 9 - 11, 2017*

# Enhancement of Driving Strategy of Electric Vehicle by Consideration of Individual Driver Intention

Meng Zhang, Karl-Falco Storm, Andreas Rausch

*Institute of Informatics, Technische Universität Clausthal*

*Julius-Albert-Straße 4, 38678 Clausthal-Zellerfeld, Germany*

*{meng.zhang, karl-falco.storm, andreas.rausch}@tu-clausthal.de*

**Summary**

Recently, the optimization of driving strategy has become a popular issue for battery electric vehicles (BEV) to increase their driving ranges. In this paper, an innovative approach for planning a driver-individual driving strategy based on self-learning algorithm is proposed. Our approach enables the driver to travel a commonly used route more energy-efficiently within his average travel time. Two self-learning algorithms respectively based on dynamic programming and Q-learning have been developed and benchmarked for BEV in this study. The proposed self-learning driving strategy can be used as target velocity for Adaptive Cruise Control (ACC).

*Keywords: optimization, energy consumption, efficiency, strategy, control system*

## 1 Introduction

The increasing energy demand has been a challenge to the limited energy resources on earth. To improve such a situation, the governments of several countries have committed themselves to reduce the greenhouse gas (GHG) emissions caused by using fossil energy sources. In the EU, emissions need to be reduced by at least 20% (from the 1990's level) [1] [2]. To achieve this goal, the European automotive manufacturers have committed to limiting their fleet emissions of GHG to 95 g/km [3]. Since combustion vehicles (CV) depend on fuel combustion and cause relatively large emissions of GHG, the development of cars using alternative drive concepts such as hybrid electric vehicles (HEV) and battery electric vehicles (BEV) becomes more important. The main advantages of electric drives are the possibility of using renewable energies (such as wind, tidal and solar power) and the possibility to recuperate energy during deceleration.

An investigation focusing on the GHG emissions of different vehicle types was performed by the non-profit organization Plug'n Drive. It acknowledged that HEVs and BEVs have substantially less GHG emissions in comparison to CVs [4]. It is worth emphasizing that the BEV got the best results regarding the GHG emission among all three vehicle concepts. However, since a BEV has limited driving range and requires long charging time [5], it is still not suitable for most use cases yet. Thus, reducing the energy consumption to improve the range of BEVs becomes an important issue. Due to this motivation, the development of optimal driving strategies as an effective solution for the issue takes a promising step forward [6].

To investigate the influence of the driving strategy on the vehicle's energy efficiency, researchers examined the driving behavior of human drivers. A noted research project focusing on the impact of driving behavior

on energy consumption and driving range stated that the power consumption of a BEV may vary about nearly 30% [7].

The driver chooses the driving strategy while driving manually, whereas it is determined by a vehicle's longitudinal control system like ACC as soon as cruise control is enabled. The fundamental ACC functionality is based on robust control theory. The driver may change parameter values for the ACC to modify its behavior, including his preferred target velocity and desired time gap (headway distance) towards preceding obstacles. Once no preceding vehicles or obstacles are present on the route, the ACC maintains the vehicle at the driver-preferred target velocity. A change in the speed limit or different route's topological profiles requires the driver to adjust the target velocity.

Either way, the driver still maintains the task of the vehicle's steering. Assuming a driver steers his vehicle with enabled ACC through a curve. In this case, the target velocity of ACC needs to be lowered to keep the lateral acceleration within safety limits. Since the required steer angle cannot be foreseen, the current velocity might be too high for this maneuver. Thus, the ACC determines an unexpected driving strategy for the driver that might force him to reclaim the vehicle's control. To overcome this problem, predictive driving strategies are increasingly considered in recent years. Based on image recognition, Car2X communication and cloud services, predictive information about the upcoming diving environment is integrated into the ACC system, like the approach realized in Volkswagen Green Driving [8].

Another example is Bosch Eco-ACC, which plans an energy-efficient deceleration strategy based on predictive route information [9]. Furthermore, Audi's predictive efficiency assistant system switches off the motor in the case of an upcoming downhill route to realize an energy-efficient coasting [10]. However, these systems only determine a local energy-optimal driving strategy based on the restricted available information about the upcoming route without considering the impact of requirements of the global energy and time demand. Against this weakness, Porsche has developed InnoDrive system, which is able to plan a global optimized driving strategy based on detailed route information [11]. When the InnoDrive is taking over the vehicle's longitudinal control, the driver also still participates in the driving process. In this case, the entire driving performance depends on the driver's driving ability and the quality of the vehicle's longitudinal driving strategy together. Due to the driver's driving ability, the driving strategy determined by the InnoDrive might be critical in the case of a sharp curve.

Porsche InnoDrive also provides the driver a limited number of operating modes, namely "sport", "eco" and "comfort". Unfortunately, they do not satisfy the diverse drivers' needs when terms of the comfort, time and energy consumption conflict. Vehicles' longitudinal control systems like the ACC or InnoDrive are generally designed as a convenience system rather than a safety system [12]. They do not engage the full braking potential of the vehicle in emergency cases. Due to the highly dynamic driving environment, a stronger deceleration than allowed might be required. In such a critical case, the system will automatically switch off and the driver has to perceive in time to reclaim the vehicle's longitudinal control. Due to the over-reliance of the ACC and a loss of vigilance of the driver, an accident risk in this case might be caused [13].

We propose a new approach to plan a driver-individual driving strategy by observing and learning the driver's preferences. Based on the sufficient knowledge about the driver's preferences recorded from past trips in a known driving environment, our approach interprets the driver intention as his preferred velocity and acceleration on the route. The planned driving strategy can be used as the target velocity for the ACC or similar vehicle's longitudinal control systems. In this study, two concepts of self-learning algorithms respectively based on dynamic programming (DP) and Q-learning (reinforcement learning) are developed to realize the approach. Both concepts are evaluated with the help of the simulated driving behavior for three scenarios: urban, extra-urban and motorway.

Since the self-learning algorithms also need the vehicle's energy consumption to evaluate the driving behavior, we use the model of a BEV's drive train as an example. For evaluation, we prove the saving potential of the vehicle's energy consumption. Thus, the algorithms are able to realize e.g. an energy-efficient braking in diverse scenarios. From a general perspective, our approach is also transferable to other vehicle concepts like HEV and CV by adapting the model of the drive train respectively. In further research, the potential of our approach for other vehicle concepts will also be investigated, as well as its safety capabilities.

The following chapters focus on our working process as well as the research results. Chapter 2 introduces the concept of the driver model and the development of the simulation environment for the self-learning algorithms. The use of a statistical driver model Move3F is described. In Chapter 3, both concepts of the self-learning algorithms are introduced. Chapter 5 gives a summary of this study and an outlook for possible further research.

# 2 Preliminaries

To evaluate the approach of a planned self-learning driving strategy, an evaluation environment based on MATLAB/Simulink was developed. As mentioned in the latter chapter, the statistical driver model Move3F was used in this study to acquire driving profiles. To investigate the consumption-saving potential of our approach in different driving scenarios, artificial velocity profiles were generated. Based on the driver model, 300 trips for each scenario including urban, extra-urban and motorway were simulated. The three routes, each having a length of about 30km, were synthetically combined for our simulation to cover a larger variety of scenarios. Additional traffic participants were not yet included, in order to minimize disturbances. Two concepts respectively based on Q-learning and DP were benchmarked with the help an evaluation environment. This chapter focuses on the introduction to the driver model and the evaluation environment.

## 2.1 Data preparation

Following the 3F methodology that has been developed at the Technische Universität Braunschweig in 1997, a generator of driving behavior had been set up called Move3F. Originally, 3F is a description method for the three factors that affect human driving behavior: the driver, the vehicle and the driving environment. By the distinction of three to four different characteristics for each of these factors, a total of 48 different driving conditions are distinguished as shown in Fig. 1 [14]. Some characteristics are more likely to be constant over the time (e.g. vehicle's physical limits), while others can more easily be predicted or assumed (e.g. course of the road) or rather be statistically (e.g. driver's driving quality). Based on their properties, each factor is represented by an individual subsystem inside the Move3F model.

To achieve the goal of a universal simulation environment, a modular architecture, including subsystems for driving behavior, driving environment and vehicle had been developed. The scheme of these subsystems is following the major axis of the 3F parameter space as shown in Fig. 1. Each subsystem can be adapted to the application of the simulation, focusing on different grades of output data. The model has been compiled on a large basis of recorded trips involving different drivers at different road, weather and daytime conditions. A large set of generated driving profiles is similar to a pool of recorded driving profiles in terms of load spectrum and the appearance of special conditions. [14]

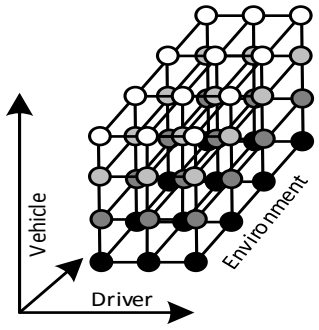


Figure 1: 3F-parameter space [14]

The vehicle model is set up as a mathematical model representing the physical properties of the corresponding drive train. As a function of the current vehicle speed, acceleration and environmental forces, the model calculates the actual energy consumption of the simulated vehicle. Furthermore, it calculates the speed and acceleration of the following simulation step by considering its actual condition plus the driver's control input (e.g. gear, accelerator and brake pedal position). The vehicle subsystem is also responsible for simulating the behavior of a real vehicle in a real environment as shown in Fig. 2. Its parameterization follows the measurement data of the corresponding real vehicle on the test bed [14].

Among others, the vehicle subsystem requires data about the slope of the road. Therefore, the next subsystem represents the driving environment. Basically, it provides information about the road profile and environment. For the driver model, the upcoming speed limit, curvature and slope are acquired from a map database to calculate a speed guidance profile of the route. It can be used universally for different drivers [14].

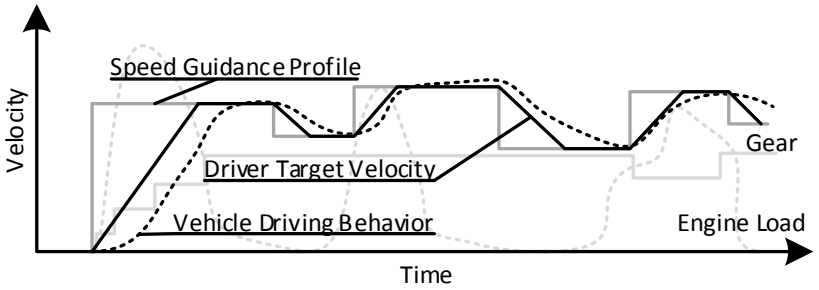


Figure 2: Vehicle model calculating the engine load as a function of vehicle speed, acceleration (and gear, for CV and HEV). Driver's target speed also shown [14]

The last subsystem relates to the driver behavior. Based on the actual course of the road, it calculates a target speed (see Fig. 3). The driver model performs different types of drivers by using different sets of parameters which are describing their behavior. For each individual driver, the parameter sets represent dynamic effects, e.g. traffic load, driver's mood and the purpose of the trip, which are then used to calculate a target velocity. [14]

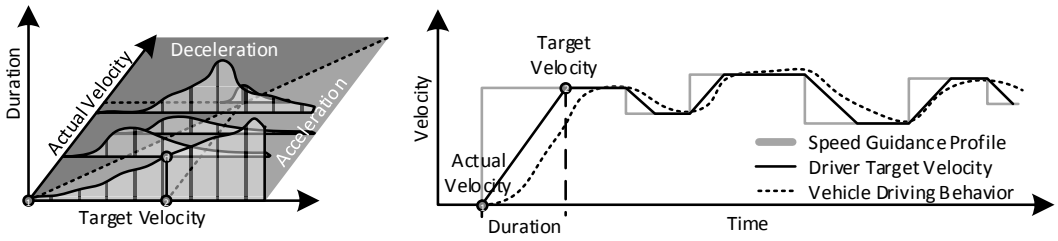


Figure 3: Selection of the acceleration and deceleration behavior [14]

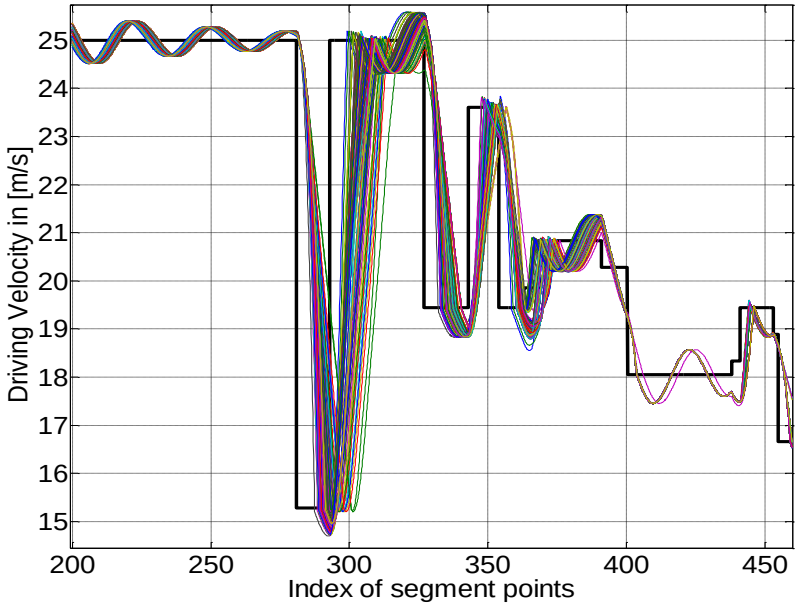


Figure 4: Sample data of driving behaviors artificially generated by Move3F

Once the target velocity is available, a second calculation process decides the position of the accelerator and brake pedal depending on the vehicle's current velocity and the target velocity. Fig. 3 shows a sample

driving behavior set, including accelerations and decelerations. Each acceleration or deceleration is planned depending on the actual difference between the current and target velocity. The corresponding duration value represents the necessary period until the driver's target velocity reaches the current speed guidance profile of the environment subsystem. In addition to the mathematical model, a statistical model modifies the target velocity to produce more realistic velocity trajectories. This procedure ensures that the generated speed profiles are not just exact copies of the observed driving behavior, but also rather distinguished in fine details (see Fig. 4). Then, the target velocity is forwarded to the vehicle subsystem in order to calculate the final velocity profile of the driving behavior. [14]

For our simulation, velocity and energy consumption have been generated with the help of Move3F. In contrast to HEV and CV, a BEV does neither need complex operating strategy respecting the drive train capabilities, nor does it need a gearbox model for automatic or manual transmission. Due to this reason, the BEV vehicle model is the first choice to compute the energy demand for the simulation. The simulation environment and its related processes are illustrated in the following section.

## 2.2 Simulation environment

Inspired by a benchmark performed in [15] using Q-learning and DP techniques to enhance the operating strategy of HEVs, we developed the process shown in Fig. 5 for our simulation. Firstly, the human driving behavior was generated for a total of 900 trips, including 300 urban, 300 extra-urban trips and 300 motorway trips for an individual driver. These behavior data were then handed over to the self-learning algorithms. The training process includes three test cases in accordance with the three scenarios. For each test case, one velocity profile was used to train the algorithm at each simulation iteration. After each training process, both algorithms planned a driving strategy for the simulation scenarios. In the last step, the energy consumption and the time demand of the latest planned driving strategy were compared to the average values as a benchmark of their saving potentials.

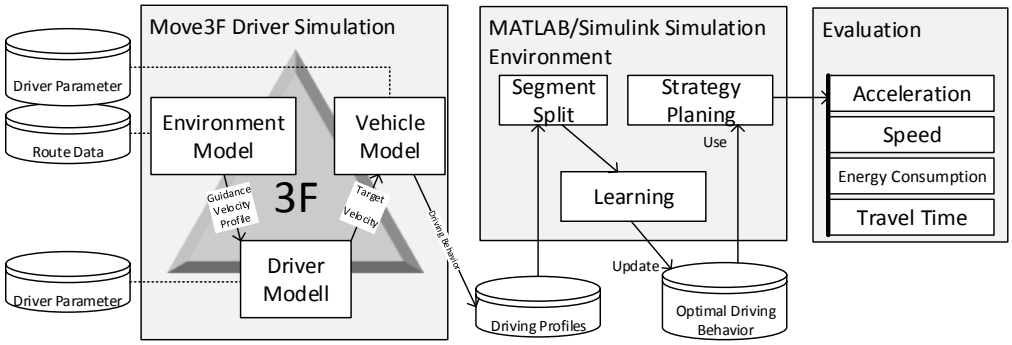


Figure 5: Process flow of the simulation

# 3 Self-learning driving strategy

The critical scenario concerning an unexpected driving strategy for a curve has been illustrated in chapter 1. To avoid this scenario, we propose an approach consisting of observing, evaluating and learning the human driving behavior while driving frequently manually on the same route. In a further step, a driver-individual driving strategy can be planned by our approach. This strategy can be used as the target velocity for the ACC.

To evaluate our approach, two self-learning algorithms respectively based on the DP and the Q-learning have been developed. For frequent trips, e.g. a driver's daily trip to work, the planned driving strategy guarantees an energy-efficient trip within the driver's average travel time.

The planning of the driving strategy in this study is abstracted as a selection of vehicle's velocities/accelerations on a segmented route (see Fig. 6). The route was segmented by discrete reference points with a fixed length of exactly one meter. For each segment, the algorithm observes the velocity/acceleration selected by the human driver. Based on the required time and energy consumption, the algorithm scores the driving behavior after the trip. Thus, an optimal driving strategy for the route can be planned by combining the highest scored driving behavior for each segment into a trajectory.

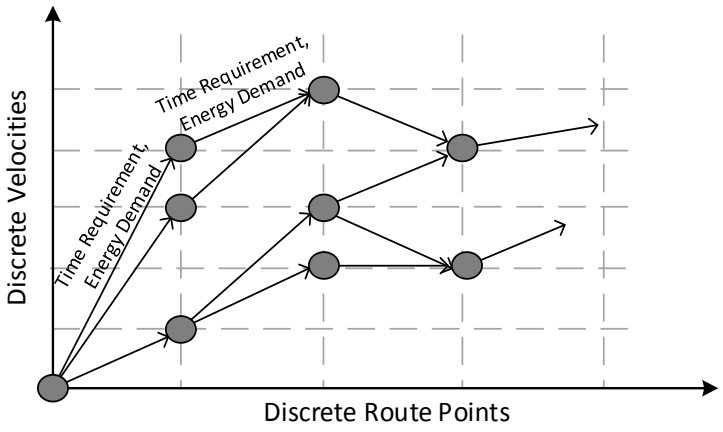


Figure 6: Decision tree of velocity selection

## 3.1 The algorithm based on dynamic programming

Firstly, dynamic programming (DP) was benchmarked. It is a typical solution to solve optimization problems satisfying Bellman's Principle of Optimality [16]. Depending on the segment points, the driving behavior is divided into a group of velocities. Thus, a weighted directed acyclic graph (DAG) is built to describe the driver's velocity selection (see Fig. 7). The DAG is comprised of nodes and edges at several layers. Each layer represents one segment points. The nodes of each layer represent observed velocities for the segment point. On every edge of the DAG, two scores $S_t$ and $S_c$ are calculated based on the required travel time and the vehicle consumption. DP consists of two working processes: learning and planning. Before learning, the scores of each edge are combined into a joint score following formula (1):

$$s = weight\_t \cdot s_t + (1 - weight\_t) \cdot s_c \tag{1}$$

All joint scores will be added to the upcoming node of the current edges. The initial node $v_0$ always has a score of zero. Other nodes are initially scored with Infinity. When the sum of the joint score of the edge and the score of the upcoming node is less than the current score of the upcoming node, the score of the node will be replaced by the sum. Otherwise, this joint score will be ignored. This process guarantees that each node only recognizes its lowest scored parent edge by saving the sum of its own score and the joint score of the edge. It can be assumed that the entire driving strategy is divided into a group of local driving strategies based on the segment points. Each local driving strategy is then represented by an edge of the DAG. Based on the approach of DP, this means that for all selected velocities only the local optimal driving strategy is learnt and saved in the knowledge base. After the learning process, the second process planning focuses on planning an optimal driving strategy by selecting the velocity value rated with the minimal score of each layer of the DAG. The DP algorithm plans the optimal driving strategy for the entire route by combing local optimal driving activities for each route segment.

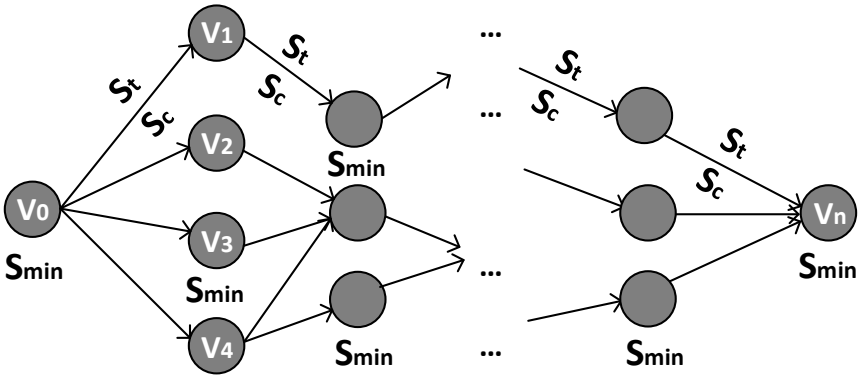


Figure 7: Weighted DAG of dynamic programming

## 3.2 The algorithm based on Q-learning

The second concept of self-learning algorithm is based on Q-learning, which is a classic model-free technique of reinforcement learning. It is very suitable to find an optimal action-selection policy for finite Markov decision processes [17]. This algorithm is usually utilized in the field of robot control. A typical example is a robot inside a labyrinth. The robot represents a learning agent, which attempts to move one step further in one of four directions up and down, left and right (representing an activity). The robot knows its position as the current state of the learning agent. After each movement, the completed activity is evaluated based on the current distance between the robot's position and the exit (as feedback) of the labyrinth (as environment). The algorithm doesn't need to model the learning agent itself. It builds an assessment process to evaluate the quality of the completed activities based on the feedback from the environment. For a positive feedback, the learning agent gets a reward (or punishment) to reinforce (or weaken) its memory of the completed activity. Once sufficient attempts have been carried out, the best route containing the minimal number of steps can possibly be found based on the rewards.

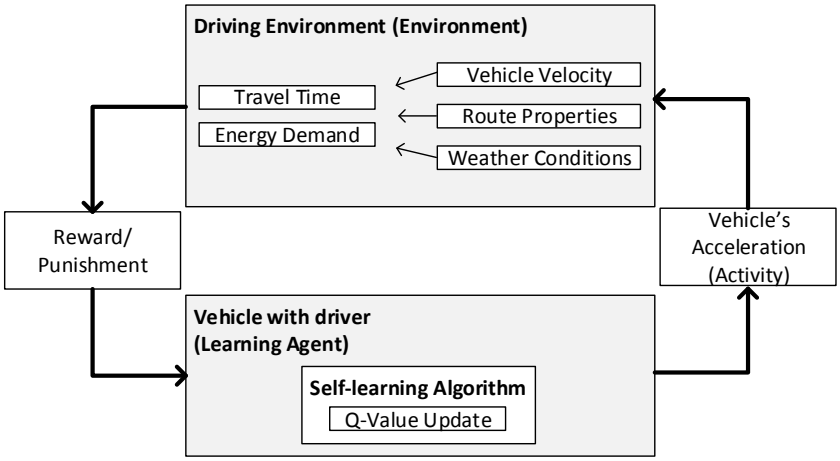


Figure 8: Process flow of the Q-learning algorithm

In our application, the learning agent is represented by the vehicle. For normal Q-learning, the learning agent attempts to perform the activity by itself. In this study, the driver is considered as an activity generator, since this algorithm is designed to learn his driving behavior. The state of the learning agent is decided by the vehicle's position (its current segment point on the route) and its velocity. The activity of the learning agent is represented by the vehicle's average acceleration for the route segment. Different accelerations and initial velocities cause different time and energy consumption to achieve the segment trip. Here, the time and energy values are used together as feedback from the environment to evaluate the completed activity of the learning agent. A Q-value is used to quantify the reward (or punishment). Fig. 8 introduces the process flow of the Q-learning algorithm.

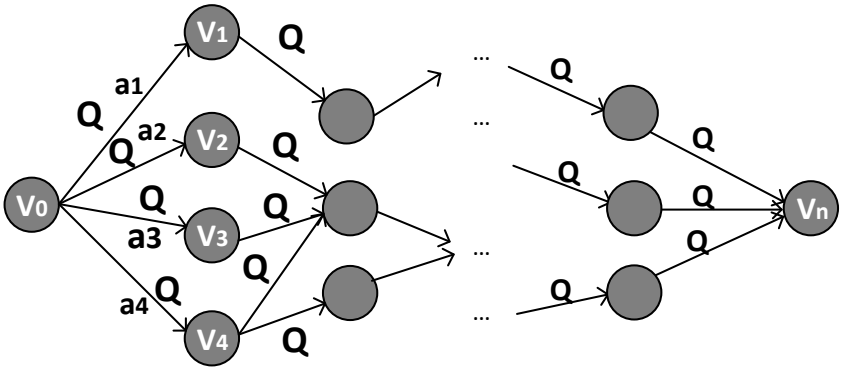


Figure 9: Weighted DAG of Q-learning

Similar to the DP algorithm, planning a driving strategy here can be abstracted as an acceleration decision problem. A weighted DAG is built up to describe the working principle of the Q-learning algorithm (see

Fig. 9). This DAG also has a multilayer structure with nodes and edges. Each edge represents an observed average acceleration for the route segment, which is scored by a Q-value.

The Q-learning algorithm also has learning and planning processes. The learning process is completed by updating the Q-values. A temporal difference learning strategy is used to reinforce better activities [17]. We define $Q(s,a)$ as the Q-value to take an action $a$ in state $s$. Initially, the Q-value of the activity is scored by a basic reward. This value is later updated based on the maximal Q-value of the best possible activity in the upcoming states. This process is mathematically represented in equation (2):

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \cdot (r + \gamma \cdot \max Q(s_{t+1}, a)) \tag{2}$$

In this equation, the parameter $\alpha \in [0\ 1]$ represents the learning rate and the parameter $\gamma \in [0\ 1]$ represents a discount factor to guarantee the convergence of the Q-value update. The parameter $r$ represents the reward of the completed activity. In our algorithm, the reward is calculated by the basic constant reward $r_{Basic}$ and the current vehicle consumption of the completed driving activity $c_{Current}$ and the driver's average consumption profile $c_{Average}$:

$$r = r_{Basic} + r_{Basic} \cdot \frac{c_{Average} - c_{Current}}{c_{Average}} \tag{3}$$

The activity is initialized with a negative value (punishment), once the driver's partial driving behavior up to the current vehicle position (the current segment point on the route) causes a longer travel time than the driver's average time profile. All updated Q-values are saved in the knowledge base of the algorithm. Based on the Q-values, the algorithm plans an optimal driving strategy by combining the driving activities (accelerations) for each route segment with maximal Q-values.

# 4 Evaluation

## 4.1 Benchmark of saving potential

As introduced in the section 2.2, the evaluation process repeats iteratively. The driving profiles of 900 trips at three different scenarios are used to investigate the self-learning algorithms. At each iteration, the current trip is used to train both algorithms. After the training process, each self-learning algorithm plans its energy-efficient driving strategy, which is then evaluated in terms of energy consumption and travel time. All the configuration parameters of both algorithms have been optimized (see Fig. 10-11). In the following sections, figures of the results are presented.

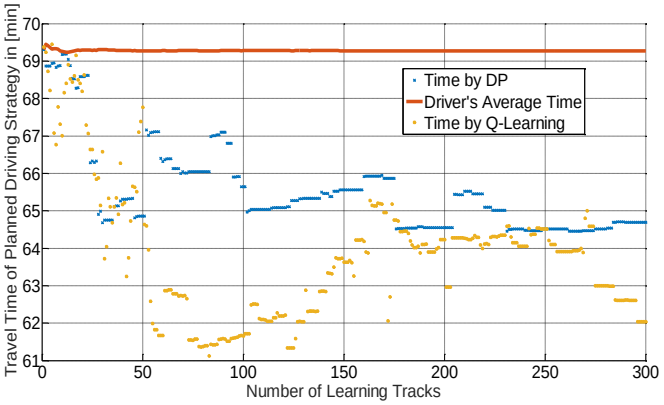


Figure 10: Travel time of planned driving strategy (extra-urban) by DP (weight_t = 0.1) and by Q-learning (α,γ= 0.001)

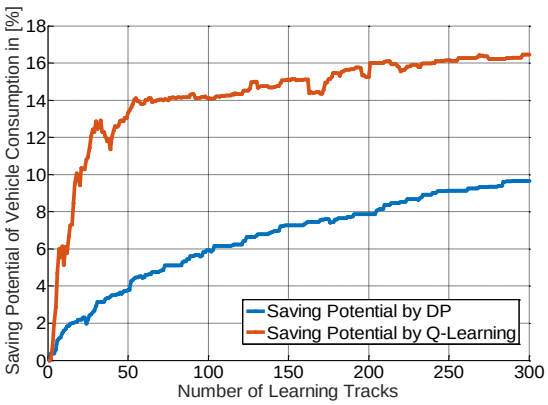


Figure 11: Consumption optimization (extra-urban) by DP (weight_t = 0.1) and by Q-learning (α,γ= 0.001)

Fig. 10 shows the caused travel time by planned extra-urban driving strategies for all 300 learning iterations. It can be seen that the required travel time (blue and yellow) is less than the driver's average travel time

(orange line). That means, both algorithms guarantee the driver to travel within his average travel time. Besides the time advantage, dynamic programming also shows its significant advantage in energy saving. As shown in Fig. 11, we can see that almost 9.8% of the energy consumption can be saved by using dynamic programming (compare to the driver's average profile).

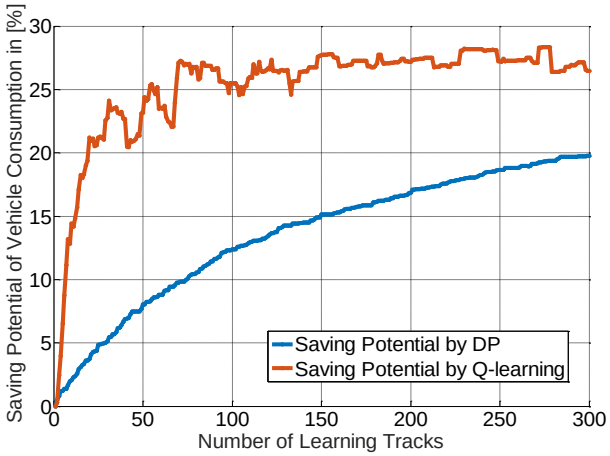


Figure 12: Consumption optimization (urban) by DP (weight_t = 0.01) and by Q-learning (α,γ= 0.001)

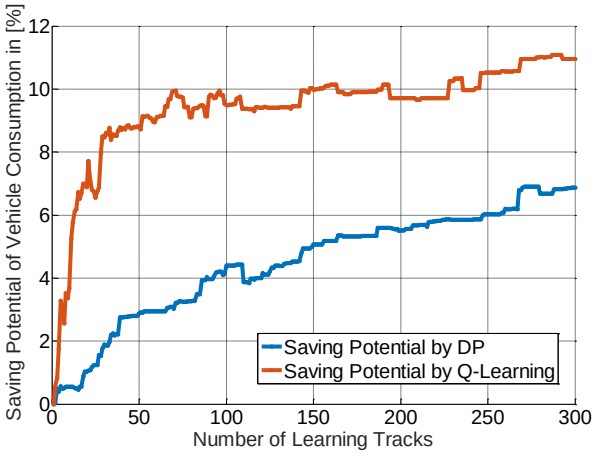


Figure 13: Consumption optimization (motorway) by DP (weight_t = 0.3) and by Q-learning (α,γ= 0.001)

The algorithm based on the Q-learning guarantees a saving potential of vehicle consumption up to about 16.5%, namely 6.7% more than the algorithm based on dynamic programming. A similar evaluation was also performed for the urban scenario. After 300 learning iterations, the algorithm based on DP achieves an energy saving potential of nearly 20% (see Fig. 12). In comparison, the Q-learning algorithm has a significant higher learning rate at the beginning. Furthermore, the Q-learning also has more saving potential in the urban scenario of about 26%. It is worth emphasizing that the planned urban driving strategy is usually not realizable for the vehicle's control system due to the occurrence of a complex traffic scenario through preceding obstacle, pedestrians, traffic lights and intersections. These influences were not yet considered in our simulation.

The last test case was performed on a motorway. As shown in Fig. 13, the DP algorithm achieved a saving potential of about 6.5%. The approach based on Q-learning shows significantly a better saving potential of about 11%. The potential depends on the quality and quantity of observed driving behavior data for training the algorithms. In case of more qualified data, the algorithms admit for further improvement.

### 4.2 Summary

In this work, two algorithms respectively based on the dynamic programming and the Q-learning were evaluated with the help of 900 artificial generated velocity profiles using the statistical driving model Move3F. For three different scenarios, both algorithms were successful in realizing energy savings. The Q-learning algorithm has a significant higher saving potential than the DP. Both algorithms guarantee that the driver achieves his travel within his average travel time.

## 5 Conclusion and outlook

### 5.1 Conclusion

The evaluation results show a significant potential of enhancing the driving strategy for ACC or similar longitudinal control systems. The DP only updates the knowledge base, once a better local driving activity is detected. Such an approach requires less computing power, but also loses some saving possibilities. Different to the DP, the Q-learning algorithm evaluates all observed driving activities. Such a concept achieves a significant higher saving potential of about 6% in average (for all three scenarios). However, the Q-learning algorithm also requires more computing power.

The planned driving strategy can be used as target velocities for the ACC or similar vehicle's longitudinal control systems. Since the driving strategy is learnt from the driver's past driving behavior, an unexpected driving strategy in the critical scenarios is filtered out (see chapter 1). The evaluation result shows that the utilization of self-learning algorithm has a significant advantage for enhancing the vehicle's control system. But such approach requires increasingly more computer power, which is unfortunately limited by the vehicle's control unit (e.g. ECU). This requirement challenges the current concept of vehicle's on-board control systems. An innovative idea is proposed in the outlook of this paper as a possible approach for solving this problem.

## 5.2 Outlook

In this study, we have evaluated the algorithm's potential of saving consumption for a BEV. It will be an interesting issue for benchmarking the potentials for other vehicle concepts like HEV or CV by using the developed algorithms in further steps.

The self-learning algorithm for planning an energy-efficient driving strategy is designed to be used as a generator of the target velocity for the ACC. From an abstract perspective, the approach in this study can be seen that a knowledge-based system is integrated into a system based on classic control theory. To realize such a goal, recent approaches have been developed based on the idea of adding a knowledge-based system into the control loop. Since the controller works in a timeframe within milliseconds, these approaches require that the knowledge-based system also works within milliseconds. This requirement challenges the computing capability of the knowledge-based system, since such a system is usually designed to process big data and requires high computing power.

The control systems work on the vehicle's control units, e.g. the ECU. Such control units have limited computing power. Thus, the original concept by adding the knowledge-based system into the control loop might cause a new problem of required computing resource. Due to the increasing complexity of the vehicle' control systems, this problem is getting worse. For this purpose, we propose here a new concept for integrating the knowledge-based system and the control system. Fig. 14 shows the structure of the new concept.

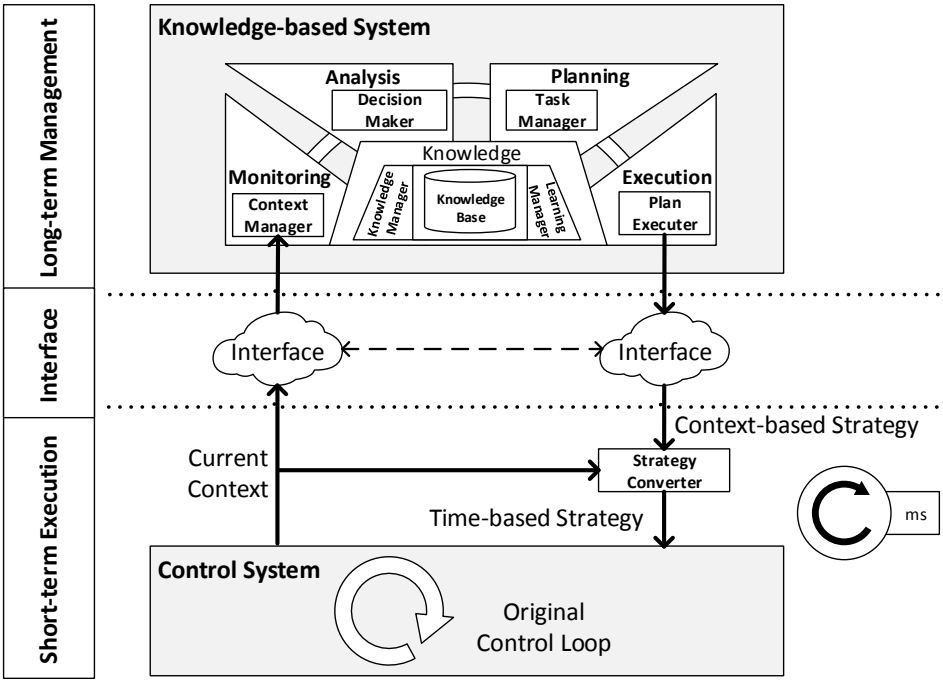


Figure 14: Future architecture concept following the MAPE-K architecture

In this concept, a software architecture MAPE-K for autonomic computing is used to describe the structure of the knowledge-based system [18]. An interface between the control system and the knowledge-based system is designed wherein the control system still clocks in the timeframe of milliseconds. The

knowledge-based system continually records the real data from the control system. Based on the context of the control system, the knowledge-based system learns the user's preferences like the driver's preferred driving behavior for the application of the ACC. All the learnt knowledge is sorted and saved in the knowledge base. The knowledge-based system then makes a decision for planning a long-term context-based strategy based on its own experiences and knowledge to fulfil the user's preferences.

The planned context-based strategy is then transferred into the control system. An additional component strategy converter (inside the control loop) processes the context-based long-term strategy into the time-based short-term strategy, which is provided to the original control system without disturbance of the control performance. Thus, the knowledge-based system doesn't work with respect to time. It realizes a context-based long-term management for the control system. The original control system is still able to work with respect to time. Once the new planned long-term strategy becomes available, the control system adjusts itself with the help of the short-term strategy of the strategy converter.

For this architecture, the knowledge-based system does not stay in the control loop and does not require high on-board computing power. With the help of external computing resource like cloud services, the both systems can work smoothly. The further development and evaluation of this concept are also included in our future work.

## Acknowledgments

The authors would like to thank Mark Schudeleit and Marian Muntau for their support regarding Move3F driver simulation. Both researchers are working at the Institute of Automotive Engineering of the Technische Universtität Braunschweig in Germany.

## References

[1] Federal Ministry for the Environment, Nature Conservation, Building and Nuclear Safety of Germany, *The EU Regulation for Reducing CO2-Emission of Passenger Cars*, http://www.bmub.bund.de/fileadmin/bmu-import/files/pdfs/allgemein/application/pdf/eu_verordnung_co2_emissionen_pkw.pdf, accessed on 2017-06-28

[2] Council of EU, *Council Decision 94/69/EC on 15 December 1993 concerning the conclusion of the United Nations Framework Convention on Climate Change*, OJ L 33 of 02 July 1994, p. 11, EU-Doc. 3 1994 D 0069

[3] S. Barsali et. Al., *A control strategy to minimize fuel consumption of series hybrid electric vehicles*, IEEE Transactions on Energy Conversion, vol. 19, 2004

[4] Plug'n Drive, *Electric Vehicles: Reducing Ontario's Greenhouse Gas Emissions*, https://plugndrive.ca/sites/default/files/Electric%20Vehicles%20-%20Reducing%20Ontario's%20Greenhouse%20Gas%20Emissions%20-%20A%20Plug'n%20Drive%20Report.pdf, accessed on 2017-06-28

[5] Akku.net, *E-Mobility auf dem Vormarsch: Vorteile, Nachteile and die Zukunft des Akku-Antriebs*, https://www.akku.net/magazin/e-mobility-vorteile-nachteile-und-die-zukunft-des-akku-antriebs/, accessed on 2017-06-28

[6] T. Radke, *Approach for Forecasting Driver's Behavior of a preceding Vehicle*, Karlsruher Publication Series Vehicle Systems Technology, Band 19, 2013

[7] C. Bingham et. Al., *Impact of driving characteristics on electric vehicle energy consumption and range, IET Intelligent Transport Systems*, vol. 6, no. 1, pp. 29-35, 2012

[8] B. Dornieden, Volkswagen AG, *Predictive Energy-efficient Vehicle's Longitudinal Control*, Development of Assistance Systems, Automobiltechnische Zeitschrift (ATZ), no. 3, 2012

[9] F. Fliehmig et. Al., *Eco-ACC for Electric and Hybrid Vehicles*, Driver Assistance System and Efficient Drives, ATZ/MTZ Reference Book, Springer Fachmedien Wiesbaden, Germany, 2015

[10] Audi Technology Portal, *Predictive efficiency assistant*, http://www.audi-technology-portal.de/en/mobility-for-the-future/audi-future-lab-mobility_en/audi-future-engines_en/predictive-efficiency-assistant, accessed on 2017-06-28

[11] M. Roth et. Al., *Porsche InnoDrive - An Innovative Approach for the Future of Driving*, 20. Aachener Kolloquium: Fahrzeug- und Motorentechnik, Aachen, Germany, 2011

[12] H. Xiong et. Al., *Driver's Adaptation to Adaptive Cruise Control, Examination of Autonomic and Manual Braking*, IEEE Transactions on Intelligent Transportation Systems, vol. 13, no. 3, Sep.2012

[13] L. Xiao et. Al., *A comprehensive review of development of adaptive cruise control systems*, Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility, 2010

[14] F. Kücükay et. Al., *Optimization of Requirements for Transmissions and Components*, Automobiltechnische Zeitschrift (ATZ), Sep. 2007

[15] X. Qi et. Al., *A Data-Driven Reinforcement Learning-Based Real-Time Energy Management System for Plug-in Hybrid Electric Vehicles with Charging Opportunities*, 95th Annual Meeting of Transportation Research Board Washington D.C., Jan. 2016

[16] R. Bellman, *Dynamic Programming*, ISBN 0-486-42809-5, Princeton, Princeton University Press, Republished 2003: Dover.

[17] E. Alpaydin, *Introduction to Machine Learning*, ISBN 0-262-01243-X, Mit Press, 2010

[18] IBM, *An architectural blueprint for autonomic computing*, http://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf, accessed on 2017-06-28

# Authors

Meng Zhang was born in Hefei, China, in 1986. In 2007, he moved to Germany as a student. He received his B. Eng. degree in Mechatronics from the Jade University of Applied Sciences in 2010. After his bachelor, he received his M. Sc. degree in Mechatronics and Information Technology from Technische Universität München. During 2012.11-2013.07, he joined TUM CREATE program and completed his master thesis with a focus on electric mobility in Singapore. Since 2014, he is a doctoral student at the cooperative doctoral program Electric Vehicle of Niedersachsen at Technische Universität Clausthal. His research focuses on the enhancement of operating strategy of electric vehicles by using machine learning algorithms.

Karl-Falco Storm was born in Uelzen, Germany in 1990. After finishing an IT apprenticeship at Volkswagen Commercial Vehicles Hannover in 2012, he enrolled at the Clausthal University of Technology. Starting in 2013, he participated in the Formula Student Electric team 'Green Voltage Racing', until he received his B. Sc. degree of Industrial Engineering in 2015. Following his thesis topic, he continued to focus on electromobility as a student researcher for the Institute for Applied Software Systems Engineering during his master studies of business informatics.

Prof. Dr. Andreas Rausch is the head of the chair for Software Systems Engineering at the Technische Universität Clausthal. Until early 2007, he was head of the chair of Software Architecture at the University of Kaiserslautern. In 2001, he obtained his doctorate at the Technische Universität München under Prof. Dr. Manfred Broy. His research in the field of software engineering focuses on software architecture, model-based software engineering and process models, with more than 70 publications worldwide. Prof. Dr. Andreas Rausch is project leader for the development of the V-Modell XT, the standard system development process model for IT systems of the Federal Republic of Germany. In addition to his research activities he participated in various commercial software projects developing large distributed systems. He is one of the four founders and shareholders of the software and consulting company 4Soft GmbH, Munich.