# Formulation and validation of a car-following model based on deep reinforcement learning

Fabian Hart[a], Ostap Okhrin[a,b], Martin Treiber[a,b,*]

[a] *TU Dresden*
[b] *Possible second address*

**Abstract**

To be written at the end

*Keywords:* reinforcement learning, car-following model, stochastic processes, string stability, validation, trajectory data

## 1. Introduction

Autonomous driving technologies are seen as promising solutions to improve road safety, where human errors account for 94% of the total accidents of Transportation National Highway Traffic Safety Administration (2015). Moreover congestion, energy consumption and emissions are intended to be reduced. But autonomous driving is a challenging task since the transportation traffic can be dynamic and unpredictable. On the way to fully autonomous driving, Advanced Driver Assistance Systems (ADAS) has been developed to solve tasks like lane-keeping, lane-changing, car-following, emergency braking, etc. Since Deep Learning methods has been demonstrated to surpass humans in certain domains, they are also adopted in the area of autonomous driving. Especially Deep Reinforcement Learning (DRL), which combines the power of Deep Learning in tackling large, complicated problems with Reinforcement Learning, has shown its potential in a broad variety of autonomous driving tasks. In Wang and Chan (2018) and Lin et al. (2020), DRL is used to guide an autonomous

*Corresponding author
*Email address:* `Martin.treiber@tu-dresden.de` (Martin Treiber)
*URL:* `www.mtreiber.de` (Martin Treiber)

vehicle safely from on-ramp to freeway. In Isele et al. (2018), Gong et al. (2020) and Tolebi et al. (2018), DRL methods are used to manage traffic of autonomous vehicles at intersections, optimizing safety and efficiency. In Wang et al. (2018), DRL is used to solve lane change maneuvers.

A further task in autonomous driving is to model the vehicle behavior under car following scenarios, where suitable accelerations has to computed in order to achieve a safe and comfortable driving. Approaches to solve this task are classical car-following models, such as the Intelligent Driver Model Treiber et al. (2000) or stochastic car-following models, such as in Treiber and Kesting (2018). Furthermore data-driven approaches use Machine Learning methods to train a car-following model based on experimental car-follower data, such as in Chong et al. (2011) or Zhou et al. (2017). Downside of this approach is that the model tries to emulate human driver behavior which still can be suboptimal.

To overcome this issue, DRL methods train non-human car-following models that can optimize metrics such as safety, efficiency and comfort. One approach is to train on scenarios, where the leading vehicle trajectory, used for training, is based on experimental data, such as in Zhu et al. (2020) or Zhu et al. (2018). Similar approaches suggest a standardized driving cycle, which functions as leading vehicle trajectory, such as Lin et al. (2019) or Yuankai et al. (2019), which uses the New European Driving Cycle. A disadvantage coming along with these approaches is, that for scenarios, which are not in the scope of the training data, the performance decreases, indicating inadequate machine learning generalization Lin et al. (2019).

Another issue of car-following models is string-stability. There are several studies focusing on dampening traffic oscillations by using a sequence of trained vehicles, such as Qu et al. (2020), Kreidieh et al. (2018) and Jiang et al. (2020).

All the mentioned DRL car-following models have three disadvantages in common: At first the acceleration range is limited in a way, that full-brakes are not considered. This results in models that are just designed for non-safety-critical scenarios. Second these models just consider car-following scenarios, while free driving or the transition between both is not reflected in the reward function. Third the trained models have just been validated on data that is similar to the training data set, so that the generalization capabilities cannot be proved.

This motivated us to design a model, which overcomes these issues. To our knowledge, no RL based car-following model has been proposed which has the following features combined:

- The proposed model considers free driving, car-following, as well as the transition between both in a way, that approaching of the leading vehicle is smooth and comfortable.
- The proposed model has a wider range of possible accelerations, which leads to a collision-free behavior also in safety-critical situations such as full-braking of the leader.
- The proposed model is trained on leading trajectories, based on an AR(1)-process, e. g. Honerkamp (1993), the parameters reflecting kinematics of real drivers. This leads to high generalization capabilities and a model usage in a broader variety of traffic scenarios.
- Different driver characteristics can be modeled by adjusting the parameters of the reward function.
- The proposed model shows string-stability.

Another feature of this work is a thorough validation of the above mentioned properties in scenarios based on both synthetic and naturalistic trajectory data, bringing the model to its limits. In all cases, the model proved to be accident free and string stable. In a further experiment the proposed model is compared to an Intelligent Driver Model, calibrated on the same data. The results indicate a better performance of the proposed model.

[short textual enumeration of the sections to come]

## 2. Model specification

The following vehicle [only named things are written in uppercase; special concepts/definitions such as Reinforcement Learning can be written both upper and lowercase; abbreviation always uppercase (RL); the "reward function" is generic, hence lowercase] is controlled by a Reinforcement Learning (RL) agent. By interaction with an environment, the RL agent optimizes a sequential decision making problem. At each time step $t$,[always comma before the main clause starts] the agent observes an environment state $s_t$ and, based on that state, selects an action $a_t$. After conducting action $a_t$, the RL agent receives a reward $r(a_t, s_t)$. The agent aims to learn an optimal state-action mapping policy $\pi$ that maximizes the expected accumulated discounted reward. [Du musst zwischen der Summe $R_t$ und den Termen $r_t$ unterscheiden. Wäre links $r_t$, wäre dies ein Gleichungssystem mit den Lösungen $\{r_t = 0 \forall t\}$ oder $\{r_t \to \infty \forall t\}$. Diese wichtige Formel verdient eine eigene Nummer.]

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}, \tag{1}$$

where $\gamma = (0, 1]$ denotes the discount factor and $\gamma^k r_{t+k}$ the expected contribution $k$ time steps ahead. The crucial elements of the RL-based [use your abbreviation] control strategy are described in detail as follows:

### *2.1. Action space*

In this study, the acceleration of the following vehicle is considered as the action of the RL agent. To enable comfortable driving and allow for a maximum braking deceleration in safety-critical situations, the acceleration is a continuous variable in the range between $a_{\min} = -9\,\mathrm{m/s^2}$ and $a_{\max} = 2\,\mathrm{m/s^2}$.[In contrast to variables or symbolic constants which are set in italic, units are set in roman (upright) font. If super- or subscripts are variables, they are set italic. If they are abbreviations (e.g., min, max) they are set upright. Multi-letter standard function names such as sin, tan, cosh or exp are written upright (use `$\sin$` etc)]

*2.2. State space*

The state space defines the observations that the following vehicle can receive from the environment. To compute an optimal acceleration, the following vehicle observes its own acceleration $a$, its own speed[ [velocity=vector, speed=abs(velocity)=scalar] $v$, the relative speed $\Delta v = v_l - v$ where $v_l$ denotes the speed of the leading vehicle [crucial since both definitions $v_l - v$ or $v - v_l$ are common] and the (bumper-to-bumper) gap $s$ to the leader [again, some use the space headway (including the leading vehicle length), and some the gap]. These observations are normalized to the range $[-1, 1]$. [How? If it is just linear translation and scaling, you need min and max values of the state variables. How to determine them?]

*2.3. Reward Function*

The goal of the RL strategy is to reduce the crash risk, while maintaining comfortable driving in non-safety-critical situations and minimizing the travel time [without that, creeping which is safe and comfortable would be the optimum strategy]. The reward function includes a set of parameters that can be adjusted to realize different driving styles. $v_{\text{des}}$ is the desired velocity that should not be exceeded. $a_{\min}$ and $a_{\max}$ are the minimum and maximum possible accelerations, as described in Section 2.1. All parameters are described in Table 1.

[first describe the element shortly, then come with the formulas] The reward function contribution at time step $t$ consists of four factors. not parts; the parts are the weighted factors, see (**??**) The first part aims to not fall below [above is OK: free flow] a reasonable distance to the leading vehicle.

[The standard normal distribution has the symbol $\Phi(.)$ and its density $\phi(.)$ or $\varphi(.)$; I prefer $\varphi$ since, then, the optical difference to the CDF $\Phi$ is greater] [Use `\left( ... \right)` to make “bigger” parentheses, brackets, braces around

Table 1: RL agent parameters and default values

| Parameter | Description | Value |
|---|---|---|
| $a_{\text{min}}$ | Minimum acceleration | $-9m/s^2$ |
| $a_{\text{max}}$ | Maximum acceleration | $2m/s^2$ |
| $b_{\text{comf}}$ | Comfortable deceleration | $-2m/s^2$ |
| $v_{\text{des}}$ | Desired velocity | $16.6m/s$ |
| $T_{\text{gap}}$ | Desired time gap to the leading vehicle | $1.5s$ |
| $s_{\text{min}}$ | Desired minimum space gap to the leading vehicle | $2m$ |
| $T_{\text{var}}$ | Time gap to describe the variance of the normal probability function (see Equation 2 - 5) | $0.7s$ |
| $T_{\text{lim}}$ | Upper time gap limit for zero reward (see Equation 2 - 5) | $15s$ |

fractions or super- and subscript quantities]

$$r_1 = \begin{cases} \frac{\varphi(s, s_{\text{opt}}, s_{\text{var}})}{\varphi(s_{\text{opt}}, s_{\text{opt}}, s_{\text{var}})}, & \text{if } s < s^* \\ \frac{\varphi(s^*, s_{\text{opt}}, s_{\text{var}})}{\varphi(s_{\text{opt}}, s_{\text{opt}}, s_{\text{var}})} \left(1 - \frac{s - s^*}{s_{\text{lim}} - s^*}\right) & \text{otherwise} \end{cases} \tag{2}$$

with

$$s_{\text{opt}} = vT_{\text{gap}} + s_{\text{min}}, \tag{3}$$

$$s_{\text{var}} = vT_{\text{var}} + 0.5s_{\text{min}}, \tag{4}$$

$$s_{\text{lim}} = vT_{\text{lim}} + 2s_{\text{min}}, \tag{5}$$

and $\varphi(x, \mu, \sigma^2)$ describing a normal probability density function. [No empty line if no new paragraph; otherwise, you get an indent that looks strange] [$s^*$ is not defined].

Figure 1 illustrates the reward function for $r_1$, containing the parameter $s_{\text{opt}}$, $s^*$ and $s_{\text{lim}}$. [Schaut man auf die Grafik, erweckt dies den Eindruck, dass

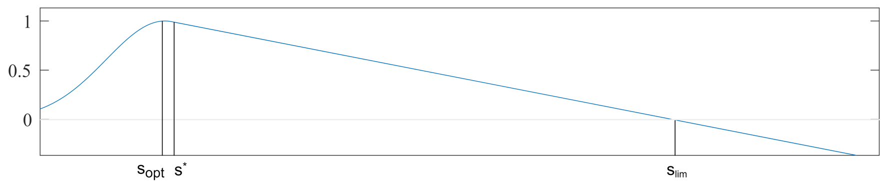

Figure 1: Factor 1 of the reward function maximizing the reward for car following with time gap $T_{\text{gap}}$

$s^*$ von $s_{\text{lim}}$ und $s_{\text{opt}}$ abhängt, genau so, dass die Funktion differenzierbar ist. Warum nicht $s_{\text{lim}} \to \infty$ setzen? Das bedeutet lediglich, dass dieser Teil der Reward-Funktion indifferent gegen sehr große Abstände ist ("es ist egal, ob der Abstand $s_{\text{opt}}$ oder $\infty$ ist, Hauptsache, er ist nicht kleiner"). Falls $v < v_{\text{des}}$ ist, wird sich der realisierte Abstand dennoch auf $s_{\text{opt}}$ einstellen. Außerdem spricht der nächste Teil der Reward-Funktion ggf auch auf größere Abstände an] The reward function is designed in a way, that for high speeds $v$ of the following vehicle the time gap between following and leading vehicle tends to $T_{\text{gap}}$, while for low speeds the distance between both tends to $s_{\text{min}}$. Different values of $T_{\text{opt}}$ result in different driving styles in a way that, for lower values of $T_{\text{opt}}$ and $s_{\text{min}}$, the driver follows more closely the leading vehicle resulting in a more aggressive driving style. The results for different values of $T_{\text{opt}}$ can be found in Section 4.4. Different functions for $s > s^*$ has been applied, but the best results regarding a smooth and comfortable approaching of the following vehicle has been reached with a linear function. Also, a high value of $T_{\text{lim}}$ results in an early deceleration and comfortable approaching. [Dafür sollte eigentlich ein Anteil $\propto (v - v_l)/s$ zuständig sein; bis jetzt kommt die "relative speed" nur quadratisch vor]

The second factor of the reward function addresses the driver's response in safety-critical situations by comparing the kinematically needed deceleration $(\Delta v)^2/(2s)$ (assuming an unchanged speed of the leader) with the comfortable deceleration $b_{\text{comf}}$,

$$r_2 = \begin{cases} \tanh\left(\frac{b_{kin}-b_{\text{comf}}}{b_{\min}}\right), & \text{if } b_{kin} > b_{\text{comf}} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

with

$$b_{kin} = \frac{\Delta v^2}{s} \tag{7}$$

[Ich habe nun nur noch "instruktive" Änderungen markiert] The argument of the tanh function with the maximum possible deceleration ($\approx 1g$ on dry roads) gives a non-dimensional measure for the seriousness of the critical situation with values near or above 1 indicating an imminent crash. [$b_{\max}$ or $-a_{\min}$]. Notice that the case distinction in (6) ensures that this term is not activated in non-critical situations as illustrated in Fig. **??**. [Das führt aber evtl zu einer "Stotterbremse" beinm Annähern an stehende Fahrzeuge, da dann $r_1$ in der Regel nicht genügend Bremskraft entwickelt]

The third factor of the reward function aims to reduce the jerk in order to achieve comfortable driving,

$$r_3 = -\left(\frac{da}{dt}\right)^2. \tag{8}$$

The fourth factor of the reward function penalizes the RL agent, if the current velocity $v$ is above the desired velocity $v_{\text{des}}$,

$$r_4 = \begin{cases} -min\left(1, (v - v_{\text{des}})^2\right), & \text{if } v > v_{\text{des}} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

The contribution of the final reward function (1) at simulation time step $t$ is the weighted sum of these factors according to

$$r_t = 0.6 r_1 + 1.1 r_2 + 0.001 r_3 + r_4, \tag{10}$$

where all the factors are evaluated at time step $t$. The weights have been found experimentally and can be optimized in future studies.

[also relate parameters to driving style attributes such as desired speed, accelerations, decelerations, desired time gap, minimum gap]

Table 2: DDPG parameter values

| Parameter | Value |
|---|---|
| Learning rate | 0.001 |
| Reward discount factor | 0.95 |
| Experience buffer length | 100000 |
| Mini batch size | 32 |
| Gaussian noise mean | 0.15 |
| Gaussian noise variance | 0.2 |
| Gaussian noise variance decay | 1e-5 |
| Number of hidden layers | 2 |
| Neurons per hidden layer | 32 |

### *2.4. RL algorithm*

In various similar control problems, the Deep Deterministic Policy Gradient (DDPG) Algorithm has been used and proven to perform well on tasks with a continuous action and state space, such as in Zhu et al. (2020), Lin et al. (2019) or Zhu et al. (2018). The original work can be found in Lillicrap et al. (2015). In order to reduce the variance of policy gradients and increase learning speed, DDPG is an actor-critic method. [Was ist eine “policy” bzw ein “policy gradient” in diesem Kontext? TRB und TRC sind nicht spezialisiert auf Deep Learning Methods, deshalb ist ein Satz Erklärung nötig] The actor determines the action, while the critic judges about the quality of the action and how the policy should be adjusted. In this study, both networks are feed-forward neural networks with two layers of hidden neurons and 32 neurons each hidden layer. All DDPG parameters are presented in Table 2.

### *2.5. Reward Function*

[Die wurden bereits in Abschnitt 2.3 spezifiziert. Parameter sollten auch dort erklärt werden. Wie man die reward function in den RL-Algorithmus einbaut, sollte in Sec 2.4 erklärt werden. Insbesondere, wie man die eigenen Bewegungen (konstante Beschleunigung?) und die Bewegungen des Leaders voraussagt, um $r_{t+k}$ für die Zukunft ($k$ Zeitschritte im Vorraus) bestimmen zu können und wie groß der prediction time horizon ist]

## 3. Model training

The training of the model comprises two important components, which have to be defined in advance: Generation of trajectories for the leading vehicle and the specification of a training episode.

### *3.1. Generating synthetic trajectories for the leading vehicle*

The leading trajectory is based on an AR(1) process, whose parameters reflect the kinematics of real leaders. The AR(1) process describes the speed of the leading vehicle and is defined as [Habe ich präzisiert]

$$v_l(t) = c + \phi v_l(t-1) + \epsilon_t \tag{11}$$

with

$$E(\epsilon_t) = 0,\ \mathrm{Var}(\epsilon_t) = \sigma,\ \mathrm{Cov}(\epsilon_t \epsilon_{t+k}) = 0 \text{ for } k \neq 0 \tag{12}$$

After reaching stationarity, this process has

$$\text{the expectation value } E(v_l) = \frac{c}{1-\phi}, \tag{13}$$

$$\text{the variance } \mathrm{Var}(v_l) = \frac{\sigma^2}{1-\phi^2}, \tag{14}$$

$$\text{the autocorrelation } \mathrm{ACF}(dt) = \phi^{dt}, \tag{15}$$

$$\text{and the correlation time } \tau = -\frac{1}{\ln \phi}, \tag{16}$$

[multi-letter special functions sin, cos, tan, ln, tanh ... have latex macros] with $d$ corresponding to the simulation step size, which is globally set to 100 ms. [use

Table 3: Assumed typical values for leading trajectories and the resulting values of the AR(1) process parameters for an update time step of 100 ms

| Real trajectory | | AR(1) process | |
|---|---|---|---|
| $v_{l,des}$ | $15\,\mathrm{m/s}$ | $\phi$ | 0.9933 |
| $a_{\text{phys}}$ | $1\,\mathrm{m/s^2}$ | $c$ | $0.05\,\mathrm{m/s}$ |
| | | $\sigma^2$ | $0.75\,\mathrm{m^2/s^2}$ |

the \unit macro applicable in text and math mode. As in the multi-letter math functions, this controls the space after it ($> 0$, less than a normal text space)]

To adjust the parameters of the AR(1) process, typical values for real leader trajectories has to be defined: With $v_{l,des}$ as the desired velocity for the leader, the mean of the AR(1) process is set to be $v_{l,des}/2$ and the standard deviation is set to be $v_{l,des}$. The acceleration $a_{\text{phys}}$ corresponds to typical physical leader accelerations. With these values and by using Equation 13 - 16, the parameters of the AR(1) process can be calculated as:

$$\phi = e^{\left(\frac{-2da_{\text{phys}}}{v_{l,des}}\right)}, \tag{17}$$

$$c = (1-\phi)\frac{v_{l,des}}{2}, \tag{18}$$

$$\sigma^2 = (1-\phi^2)\frac{v_{l,des}^2}{4}. \tag{19}$$

The assumed typical values for $v_{l,des}$ and $a_{\text{phys}}$ as well as the resulting values of the AR(1) process parameters can be found in Table 3.

Figure 2 shows an example trajectory of the leading vehicle based on the AR(1) process using the parameters of Table 3. If the velocity exceeds the defined range of $[0, v_{l,des}]$, it is manually set to the range limits.

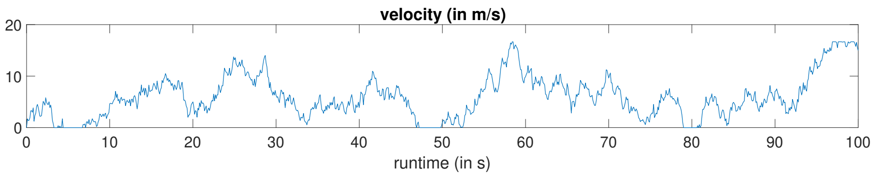

Figure 2: Example of a leading trajectory based on the parametrized AR1 process used to train the RL agent

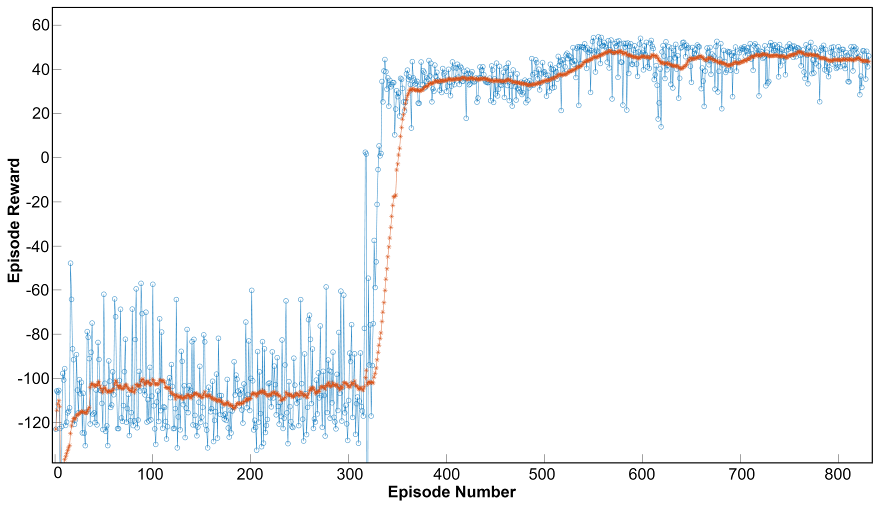

Figure 3: RL training process, one episode contains 500 steps. Shown are the reward values of the individual episodes (blue) and a 50-step asymmetric moving average [Im Gegensatz zum exponential moving average ist der normale moving average standardmäßig symmetrisch, enthält also Vergangenheit und Zukunft.]

### *3.2. Training episode definition*

To train the RL agent, a training episode has to be defined. One episode has a simulation time of 50 s with a step size of $d = 100\,\mathrm{ms}$ resulting in an episode length of 500 steps.Reine Zahlen (aber nicht das Minus-Zeichen!) sind im Text- und Math-Modus gleich. Ebenso wie Einheiten ohne Exponent benötigen positive Zahlen keine Dollars, also 500 steps oder 50 s mit oder ohne $ OK The initial speeds of the following and leading vehicles are randomly set in the range $[0, v_{\mathrm{des}}]$ and $[0, v_{l,des}]$, respectively. [“respectively” kommt immer, mit Komma abgetrennt, nach der Aufzählung, auch bei zwei items] The initial space gap between both is set to 120 m.

### *3.3. Results of the RL training process*

Figure 3 shows an example of the training process. The red line shows the moving average reward of the last 30 episodes. After approximately 570 episodes, the maximum average reward has been reached. Once the saturation has been confirmed at an episode count of 850, the learning process has been stopped. [Das Maximum war doch schon bei 570. Warum saturation erst bei 850? Ich habe es so umformuliert wie ich denke, dass der Algo läuft. Übrigens sollte man den Sprung nach etwa 350 Episoden erklären. Ist das der typische Fall? Falls ja, kann es passieren, dass der Algo vorher stoppt, da er nicht mehr denkt, dass noch ein Sprung kommt?]

## 4. Validation

The goal is not to minimize some error measure as in usual calibration/validation but to check if the driving style is safe, effective, and comfortable. The RL strategy is evaluated with respect to these metrics in different driving scenarios, described in the following.

### *4.1. Response to an external leading vehicle speed profile*

The first scenario is designed in order to evaluate the transition between free driving and car-following as well as the follower’s behavior in safety-critical situations. Figure 4 shows a driving scenario with an artificial external profile for the leading vehicle speed. The initial gap between follower and leader is 200 meters, which refers to a free driving scenario. The follower accelerates with $a_{\max} = 2m/s^2$ until the desired speed $v_{\text{des}} = 16.6m/s$ is reached and approaches the standing leading vehicle. When the gap between both drops below 90 meters, the follower starts to decelerate with a maximum deceleration of approximately $b_{\text{comf}} = 2\,\text{m/s}^2$ (transition between free driving and car-following) and comes to a standstill with a final gap of approximately $s_{\min} = 2\,\text{m}$. Afterwards ($t = 27\,\text{s}$), the leading vehicle accelerates to a speed that is below the desired speed of the follower before performing a maximum braking maneuver ($a = -9\,\text{m/s}^2$) to a full stop ($t = 40\,\text{s}$). [Die Beschleunigung ist *nicht* random]. At the time of the start of the emergency braking, the follower has nearly reached a steady following mode at the desired space gap $s = s_0 + v_l T$. While this gap makes it impossible to keep the deceleration in the comfortable range without a rear-end collision, the follower makes the best of the situation by braking smoothly with a maximum deceleration of $-a \approx 5\,\text{m/s}^2$. The transition between different accelerations happens in a comfortable way reducing the resulting jerk. Only at the beginning ($t = 39\,\text{s}$) where the situation is really critical, the jerk $\mathrm{d}a/\mathrm{d}t$ exceeds the comfortable range $\pm 1.5\,\text{m/s}^3$. Afterwards, the leader performs a series of non-critical acceleration and deceleration maneuvers and the follower tries to follow the leader at the speed dependent desired space gap $s_0 + v_l T$ while simultaneously smoothing the leader's speed profile.

## *4.2. String stability*

The second validation scenario, shown in Figure 6, consists of a leader based on the AR(1) process that is followed by five vehicles, each controlled by the trained RL agent. The results show that traffic oscillations can effectively be dampened with a sequence of trained followers, even if the leader shows large outliers in acceleration. Figure 5 illustrates the difference of accelerations between leader and the followers (blue bars). The last follower shows the lowest variance of acceleration, which demonstrates the ability of the RL agent to flat-

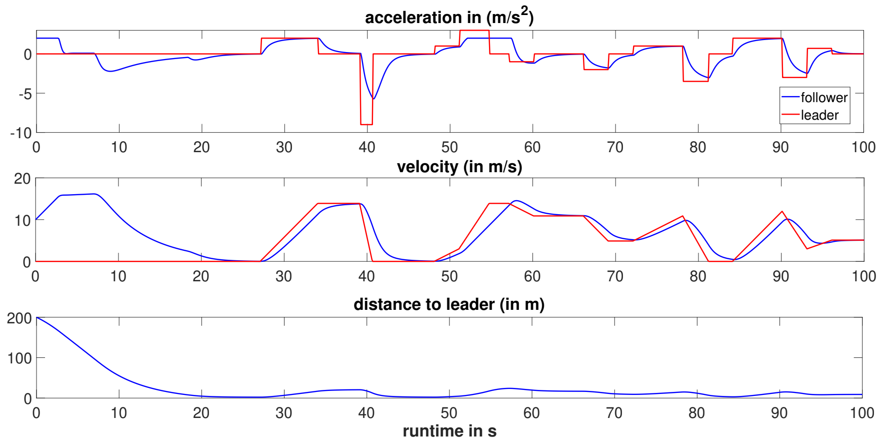

Figure 4: Response to an external leading vehicle speed profile [runtime → time, da sich runtime so nach Computerlaufzeit anhört, distance → gap. Dort sollte man die Skala nur bis 50 m gehen lassen, da man sonst fast nichts sieht, insbesondere nicht, ob $s_{\text{des}}$ annähernd eingehalten wird.]

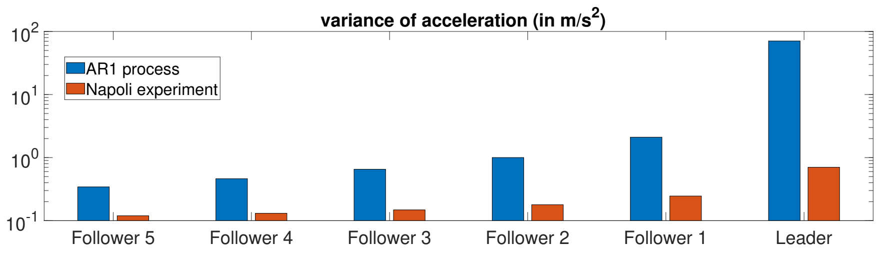

Figure 5: Comparison of the acceleration variance between leader and follower for a leader controlled by AR(1) (blue bars) and the leading vehicle of the Napoli experiment (red). [Die Varianz hat die Einheit $\mathrm{m}^2/\mathrm{s}^4$ (Korrektur in der Grafik!) In der Tat stimmt der blaue Balken des Leaders, der eine Höhe von $\sigma^2/d^2 = 75\,\mathrm{m}^2/\mathrm{s}^4$ haben sollte]

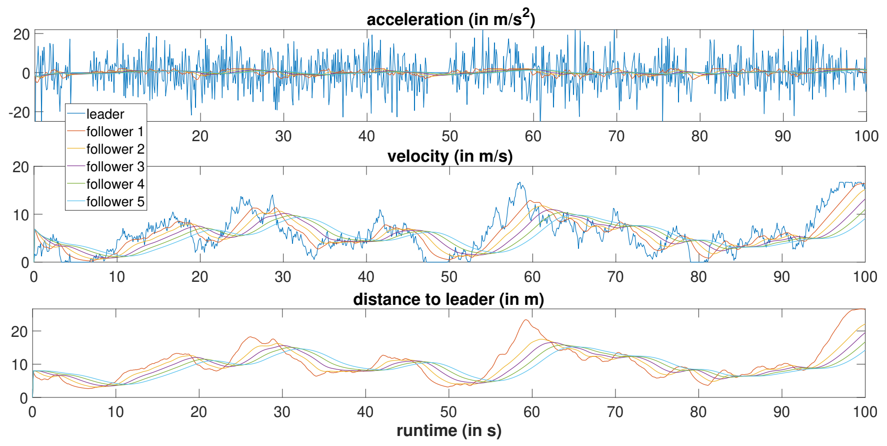

Figure 6: Response to a leader trajectory based on a AR(1) process

ten the speed profile, to dampen oscillations and thus to increase comfort and reduce fuel consumption and emissions.

## *4.3. Response to a real leader trajectory*

In a further scenario, the abilities of the RL strategy are evaluated with a real leader trajectory (Fig. 7) [Die Tilde im Latex Quelltext bedeutet ein reserviertes Leerzeichen, bei dem kein Zeilenumbruch stattfinden darf]. This trajectory comes from platoon driving experiments in Napoli where high-precision distance data between the platoon vehicles were obtained (reference to Punzo et al.). [wdh gestrichen] Similar to the experiment from Section 4.2, string stability and the reduction of the acceleration variance, shown by the red bars in Figure 5, is demonstrated. At time $t = 140s$ the leader stands still, and it can be observed, that all following vehicles are keeping the minimum distance $s_{\min}$ to the leader.

## *4.4. Response of different driver characteristics*

As mentioned in Section 2.3, different driving styles can be achieved by adjusting the parameters of the reward function. Three RL agents has been trained on a reward function, that differs in the desired time gap $T_{\text{gap}}$ between

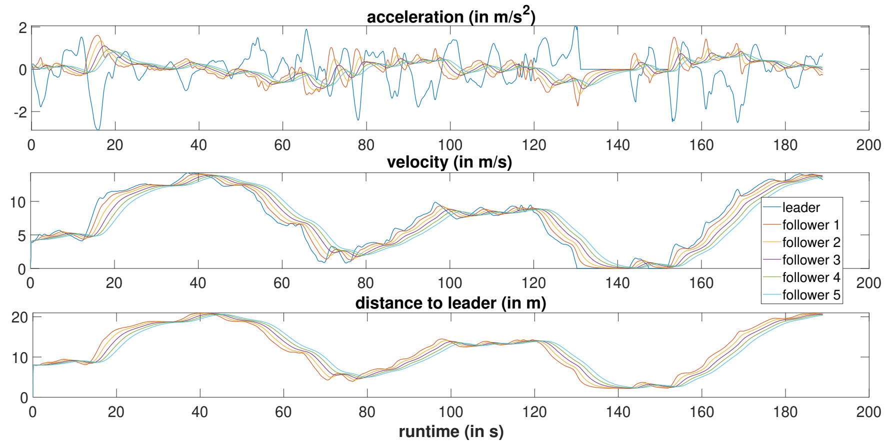

Figure 7: Response to a real leader trajectory Die leader acceleration hat das falsche Vorzeichen bzw passt für $t > 160\,\mathrm{s}$ überhaupt nicht!

following and leading vehicle ($T_{gap,1} = 1.0s$, $T_{gap,2} = 1.5s$, $T_{gap,3} = 2.0s$). Figure 8 shows the result of these agents, following the real leader trajectory from Napoli. It can be observed, that a lower value for $T_{\text{gap}}$ results in closer driving to the leader, higher accelerations and decelerations and thus in a more aggressive driving behavior.

### *4.5. Cross validation with the Intelligent Driver Model*

To compare the performance of the trained RL agent with a classical car-following model, the RL agent and an IDM are calibrated on the same set of parameters. These parameters are obtained from one and the same follower from the Napoli data set (reference) and can be found in Table 4. The IDM is formulated as

$$a = a_{\max}\left(1 - \left(\frac{v}{v_{\text{des}}}\right)^4 - \left(\frac{s^*\left(v, \Delta v\right)}{s}\right)^2\right), \tag{20}$$

with

$$s^*\left(v, \Delta v\right) = s_{\min} + vT_{\text{gap}} + \frac{v\Delta v}{2\sqrt{a_{\max}b_{\text{comf}}}}, \tag{21}$$

$$\Delta v = v - v_l \tag{22}$$

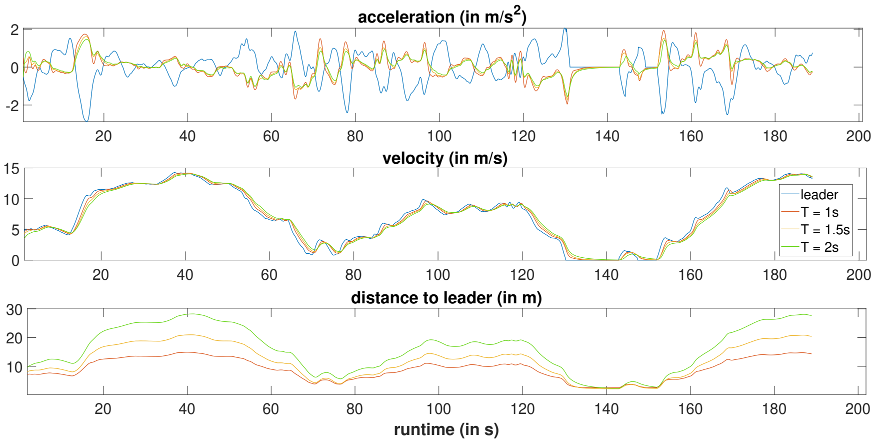

Figure 8: Impact of different parametrized RL agents on driving behavior

Table 4: Parameters obtained from experimental data and used to calibrate the RL agent and the IDM

| Parameter | Value |
|---|---|
| $T_{\text{gap}}$ | $0.83s$ |
| $s_{\text{min}}$ | $4.90m$ |
| $a_{\text{max}}$ | $4.32m/s^2$ |
| $b_{\text{comf}}$ | $2.34m/s^2$ |
| $v_{\text{des}}$ | $33.73m/s$ |

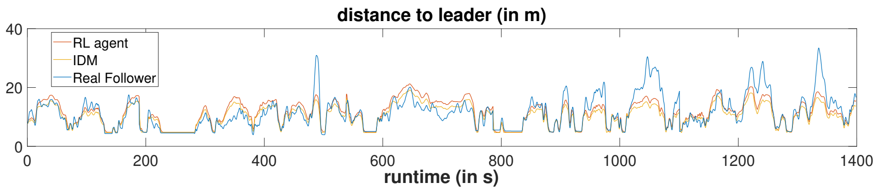

Figure 9: Comparison between IDM and RL agent, calibrated on the same set of parameters

Table 5: Comparison between calibrated RL agent and IDM for accumulated Reward and Goodness-of-Fit Function $SSE(log(s))$

| | RL agent | IDM |
|---|---|---|
| $SSE(log(s))$ | 389.10 | 418.05 |
| Accumulated Reward | $8.23 \times 10^3$ | $8.08 \times 10^3$ |

Figure 9 shows the results for: First, the RL agent, calibrated on the real follower data. Second, the IDM, calibrated on the follower data. And third, the real follower of the Napoli experiment. To evaluate the performance, for both approaches the respective objective function has been computed. The objective function of the RL agent correspondences to the reward function, while the Goodness-of-Fit Function $SSE(s)$ defines the objective function of the IDM. A comparison between RL agent and IDM for both, the reward function and the Goodness-of-Fit Function, is shown in Table 5. For both objectives the RL agent shows a better performance.

## 5. Conclusion/Discussion

evaluation of safety and comfort, comparison to IDM

U. D. of Transportation National Highway Traffic Safety Administration, Critical reasons for crashes investigated in the national motor vehicle crash causation survey, NHTSA, Washington, DC, USA (2015).

P. Wang, C.-Y. Chan, Autonomous ramp merge maneuver based on reinforcement learning with continuous action space (2018).

Y. Lin, J. McPhee, N. Azad, Anti-jerk on-ramp merging using deep reinforcement learning, 2020, pp. 7–14. doi:`10.1109/IV47402.2020.9304647`.

D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, K. Fujimura, Navigating occluded intersections with autonomous vehicles using deep reinforcement learning, 2018, pp. 2034–2039. doi:`10.1109/ICRA.2018.8461233`.

Y. Gong, M. Abdel-Aty, J. Yuan, Q. Cai, Multi-objective reinforcement learning approach for improving safety at intersections with adaptive traffic signal control, Accident Analysis & Prevention 144 (2020) 105655.

G. Tolebi, N. S. Dairbekov, D. Kurmankhojayev, R. Mussabayev, Reinforcement learning intersection controller, in: 2018 14th International Conference on Electronics Computer and Computation (ICECCO), 2018, pp. 206–212. doi:`10.1109/ICECCO.2018.8634692`.

P. Wang, C. Chan, A. de La Fortelle, A reinforcement learning based approach for automated lane change maneuvers, in: 2018 IEEE Intelligent Vehicles Symposium (IV), 2018, pp. 1379–1384. doi:`10.1109/IVS.2018.8500556`.

M. Treiber, A. Hennecke, D. Helbing, Congested traffic states in empirical observations and microscopic simulations, Physical Review E 62 (2000) 1805–1824.

M. Treiber, A. Kesting, The intelligent driver model with stochasticity – new insights into traffic flow oscillations, Transportation Research Part B: Methodological 117 (2018) 613 – 623. TRB:ISTTT-22.

L. Chong, M. M. Abbas, A. Medina, Simulation of driver behavior with agent-based back-propagation neural network, Transportation Research Record 2249 (2011) 44 – 51.

M. Zhou, X. Qu, X. Li, A recurrent neural network based microscopic car following model to predict traffic oscillation, Transportation Research Part C: Emerging Technologies 84 (2017) 245–264.

M. Zhu, Y. Wang, Z. Pu, J. Hu, X. Wang, R. Ke, Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving, Transportation Research Part C: Emerging Technologies 117 (2020) 102662.

M. Zhu, X. Wang, Y. Wang, Human-like autonomous car-following model with deep reinforcement learning, Transportation Research Part C: Emerging Technologies 97 (2018) 348–368.

Y. Lin, J. McPhee, N. Azad, Comparison of deep reinforcement learning and model predictive control for adaptive cruise control (2019).

W. Yuankai, H. Tan, J. Peng, B. Ran, A deep reinforcement learning based car following model for electric vehicle, Smart City Application 2 (2019).

X. Qu, Y. Yu, M. Zhou, C.-T. Lin, X. Wang, Jointly dampening traffic oscillations and improving energy consumption with electric, connected and automated vehicles: a reinforcement learning based approach, Applied Energy 257 (2020) 114030.

A. R. Kreidieh, C. Wu, A. M. Bayen, Dissipating stop-and-go waves in closed and open networks via deep reinforcement learning, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018, pp. 1475–1480. doi:10.1109/ITSC.2018.8569485.

L. Jiang, Y. Xie, D. Chen, T. Li, N. Evans, Dampen the stop-and-go traffic with connected and automated vehicles- a deep reinforcement learning approach, 2020.

J. Honerkamp, Stochastic dynamical systems: concepts, numerical methods, data analysis, John Wiley & Sons, 1993.

T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, CoRR (2015).