2017 16th IEEE International Conference on Machine Learning and Applications

An Evolutionary Approach to General-Purpose Automated Speed and Lane Change Behavior

Carl-Johan Hoel*†, Mattias Wahde*, Krister Wolff*
*Chalmers University of Technology, 412 96 Göteborg, Sweden
†Volvo Group Trucks Technology, 405 08 Göteborg, Sweden
Email: {carl-johan.hoel, mattias.wahde, krister.wolff}@chalmers.se

***Abstract*—This paper introduces a method for automatically training a general-purpose driver model, applied to the case of a truck-trailer combination. A genetic algorithm is used to optimize a structure of rules and actions, and their parameters, to achieve the desired driving behavior. The training is carried out in a simulated environment, using a two-stage process. The method is then applied to a highway driving case, where it is shown that it generates a model that matches or surpasses the performance of a commonly used reference model. Furthermore, the generality of the model is demonstrated by applying it to an overtaking situation on a rural road with oncoming traffic.**

I. Introduction

Technology for limited automation of vehicles has been available for some time. Different functions exist to automate specific driving tasks, such as adaptive cruise control, lane keeping systems, automated parallel parking etc. One of the more mature areas of automated driving is highway driving. Much of the complexity from e.g. city driving is here reduced, and during normal operation (excluding special events such as road work etc.) the task is limited to adapt the vehicle speed, follow the intended lane, and carry out lane changes.

Many algorithms for speed control exist; for an overview, see e.g. [1]. One commonly used model for speed control is the *Intelligent driver model* (IDM) [2], which is a simple but robust model that considers the relative distance to, and speed of, the preceding vehicle. In order to decide whether or not a lane change is feasible, different gap acceptance models are often used; see, for example, [3] or [4]. Another algorithm, *Minimize overall braking induced by lane changes* (MOBIL) [5], tries to minimize the longitudinal acceleration of all participants in a traffic situation. A so called politeness factor determines the level of priority of the vehicle under consideration (henceforth referred to as the *ego vehicle*) compared to that of surrounding vehicles. There is also a variety of more advanced algorithms for lane changing, taking into account the risk and utility associated with a lane change [6] or solving the problem as a partially observable Markov decision process [7].

However, most existing methods for automated driving, for example the methods mentioned above, are tailored to handle a specific case, such as highway driving. As soon as a different setting is considered, e.g. driving on a rural road with oncoming traffic instead of a highway, an entirely different method may be required. Moreover, human drivers do not in general drive in a continuously error-minimizing way, as is often assumed in methods based on classical control theory. Instead, human driving is a more complex process involving several cognitive tasks such as neuro-muscular control, monitoring, and decision-making, see [8] and [9].

Therefore, in this paper an alternative method for automated driving is introduced, in which a genetic algorithm (GA) with length-varying chromosomes is used for optimizing a set of rules in simulation, in order to form a complete automated driver model for a given situation, although the number of available lanes, the speed of the vehicles, the presence or absence of oncoming traffic, etc. can differ. The GA is applied in an open-ended manner, optimizing both the structure (i.e. the number of rules, as well as the types of rules used) and the parameters of the set of rules constituting the driver model. For details of GAs in general, see e.g. [10]

The main strength of the proposed method is its ability for generalization: It is shown that this approach is able to generate a driver model for highway driving with a performance that matches the combination of the IDM and the MOBIL model. Moreover, it is shown that a given rule set can, with some tuning, be applied in an entirely different setting, namely (in this case) driving on a rural road with oncoming traffic.

This paper is organized as follows: Sect. II gives an overview of the IDM and the MOBIL model, which are used for comparison, and describes how the simulations were set up. The genetic algorithm is outlined in Sect. III. Next, in Sect. IV, the results are presented, followed by a discussion in Sect. V. Finally the conclusions are given in Sect. VI.

II. Simulation environment

As a baseline for comparison, a reference driver model, combined from the IDM and MOBIL model, was used for longitudinal control and for lane change decisions. Highway traffic was simulated, which included frequent speed changes to accelerate the training process. This section describes the reference model and how the simulations were set up.

A. Reference model

The IDM [2] has often been used for driver modeling, driver assistance systems, and traffic simulations. It is a longitudinal model of the ego vehicle's speed, v, according to

$$\dot{v} = a\left(1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*(v,\Delta v)}{s}\right)^2\right), \tag{1}$$

$$s^*(v,\Delta v) = s_0 + vT + v\Delta v/(2\sqrt{ab}). \tag{2}$$

Here v is the speed of the ego vehicle, Δv is the speed difference between the ego vehicle and the vehicle in front (approach rate), and s is the distance to the veh

0-7695-6321-X/17/31.00 ©2017 IEEE
DOI 10.1109/ICMLA.2017.00-70

the ego vehicle. These quantities act as the inputs to the model, whereas the tuning parameters are the minimum gap distance, s_0, the safe time headway, T, the maximal acceleration, a, the desired deceleration, b, the acceleration exponent δ, and the desired free speed, v_0.

The MOBIL model [5] uses incentives and safety criteria to decide if a lane change should take place. The goal of the model is to minimize the overall negative acceleration of all participants in a traffic situation. In order to estimate the future accelerations of both the ego vehicle and the surrounding vehicles, when choosing to stay in lane or to change lane, the IDM is applied to the different cases.

Two criteria need to be fulfilled to perform a lane change. The first requires that the deceleration of the trailing vehicle in the target lane, $\tilde{a}_n$, must not exceed a safe limit, b_{safe}, i.e. $\tilde{a}_{\text{n}} > -b_{\text{safe}}$. The second criterion states that the incentive to make a lane change should be larger than a specific threshold. The incentive is calculated from the induced accelerations of the surrounding vehicles both when performing, and when not performing, a lane change. This criterion is described by

$$\tilde{a}_{\text{e}} - a_{\text{e}} + p\left((\tilde{a}_{\text{n}} - a_{\text{n}}) + (\tilde{a}_{\text{o}} - a_{\text{o}})\right) > a_{\text{th}}, \qquad (3)$$

where $\tilde{a}_{\text{e}}$, $\tilde{a}_{\text{n}}$ and $\tilde{a}_{\text{o}}$ are the accelerations of the ego vehicle, the trailing vehicle in the target lane, and the trailing vehicle in the current lane, respectively, in the case where the lane change is carried out. Similarly, a_{e}, a_{n} and a_{o} are the accelerations of the ego vehicle, the trailing vehicle in the target lane, and the trailing vehicle in the current lane, respectively, in the case where the lane change does not occur. The politeness factor p controls how much the effect on the surrounding vehicles is considered. Furthermore, the threshold a_{th} decides how advantageous a potential lane change should be in order to be executed. In summary, the model weighs the own advantage (acceleration gain) versus the sum of the disadvantages for other vehicles (acceleration losses).

The same criteria are applied both to the left and right lanes, if they are both available. In cases where the criteria are fulfilled both to the left and right, the most advantageous option, in the form of the highest acceleration gain, is chosen.

In this study, the parameters (given in Table I) from the original papers [2] and [5] were used.

B. Traffic simulation

The main case where the method presented in this paper was applied involved highway driving. More specifically, a straight three-lane highway was used, with U.S. passing rules, i.e. such that it was allowed to overtake a vehicle both on the left side and the right side. Each simulation consisted of one ego vehicle to be controlled and 9 surrounding vehicles, which followed the IDM model longitudinally and stayed in their lanes. In all cases, the ego vehicle consisted of a 16.5 m long truck-semitrailer combination, whereas the surrounding vehicles were normal passenger cars.

Normal highway driving often involves traffic with more or less constant speeds and small accelerations. However, occasionally, vehicles have to brake hard, or even carry out emergency braking to avoid collisions. In order to train a model that can handle situations like these using machine learning, naturally such events must be included in the training simulations. All the vehicles in the simulation were assigned different desired speed trajectories, some examples of which are given in Fig. 1. With the aim of speeding up the training of the model, traffic was simulated with frequent speed changes.

TABLE I
IDM AND MOBIL MODEL PARAMETERS.

Minimum gap distance, s_0	2.0 m
Safe time headway, T	1.6 s
Maximal acceleration, a	0.7 m/s^2
Desired deceleration, b	1.7 m/s^2
Acceleration exponent, δ	4
Politeness factor, p	1
Changing threshold, a_{th}	0.1 m/s^2
Maximum safe deceleration, b_{safe}	4.0 m/s^2

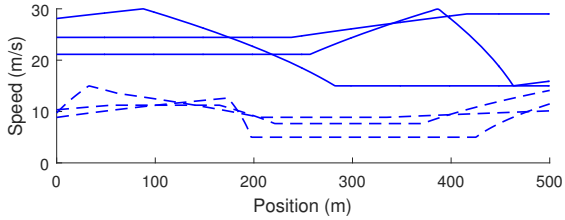

Fig. 1. Example of six different randomly generated speed trajectories, defined for different positions along the highway. The solid lines are fast trajectories, applied to vehicles starting behind the ego vehicle, whereas the dashed lines are slow trajectories, applied to vehicles starting in front of the ego vehicle.

The ego vehicle always started in the middle lane. Then the 9 surrounding vehicles were randomly positioned around it, within a distance d_{long} longitudinally, set to 150 m. All vehicles were separated with a minimum inter-vehicle distance d_Δ, set to 20 m. The surrounding vehicles were assigned desired speed trajectories, which were randomly generated by the following procedure: If the vehicle was placed in front of the ego vehicle, an initial speed was randomly chosen in the range $[v^+_{\text{min}}, v^+_{\text{max}}]$, here taken as $[5, 15]$ m/s. If instead it was placed behind the ego vehicle, the initial speed was chosen in the range $[v^-_{\text{min}}, v^-_{\text{max}}]$, here taken as $[15, 30]$ m/s. Then, with a sampling time of 1 s, a choice was made (with equal probability) either to keep the speed constant or to apply a constant acceleration. With equal probability, a positive or negative acceleration was used. The acceleration was drawn from a normal distribution with mean 0 and variance σ m/s^2. In case of positive accelerations, negative values were rejected, and vice versa for negative accelerations. σ was equal to 1 for positive accelerations, and 5 for negative accelerations. The modulus of the acceleration was limited to 2σ. This acceleration was kept constant for a randomly selected duration in the range $[t_{\text{min}}, t_{\text{max}}]$, set to $[2, 20]$ s for positive accelerations and $[0.4, 4]$ s for negative accelerations.

Furthermore, for vehicles initialized in front of the ego vehicle, the speed was limited to the range $[v^+_{\text{min}}, v^+_{\text{max}}]$ (see above), and for vehicles initialized behind the the ego vehicle, to the range $[v^-_{\text{min}}, v^-_{\text{max}}]$. The initial desired ego vehicle speed, $v^{\text{ego}}_{\text{init}}$, and maximum ego vehicle speed, $v^{\text{ego}}_{\text{max}}$, were set to 15 m/s and 20 m/s respectively. Finally, scenarios where any two vehicles were placed too close together with a large speed difference, thus causing an unavoidable collision, were deleted.

An example of an initial traffic situation is shown in Fig. 2.

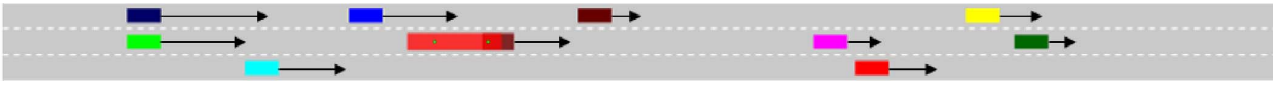

Fig. 2. Example of an initial traffic situation. The ego vehicle (a truck-trailer combination) is shown in red, in the middle lane. The arrows represent the velocities of the vehicles.

TABLE II
LATERAL CONTROL PARAMETERS.

Distance to near point	5 m
Distance to far point	100 m
Proportional gain far point, k_f	20
Proportional gain near point, k_n	9
Integral gain near point, k_I	10 s^{-1}

C. Vehicle and lateral control model

For each vehicle a lane-following controller was implemented. A two-point visual control model [11] was used,

$$\dot{\delta} = k_f \dot{\theta}_f + k_n \dot{\theta}_n + k_I \theta_n, \tag{4}$$

where $\dot{\delta}$ is the steering angle, and θ_n and θ_f are the perceived angles to a near and far point in the intended lane, respectively. k_f, k_n and k_I are control parameters. The parameter values are given in Table II.

The vehicles were modeled using a simple kinematic model. Regardless of the desired acceleration (see Sect. II-B), the actual acceleration of the vehicles was limited to the range $[a^\mathrm{M}_\mathrm{min}, a^\mathrm{M}_\mathrm{max}]$, here taken as $[-10, 2]$ m/s^2, and the maximum speed was limited to $v^\mathrm{M}_\mathrm{max} = 30$ m/s.

III. GENETIC ALGORITHM

As mentioned in Sect. I, a GA with length-varying chromosomes was used to train a rule-based driver model in simulation. GAs are suitable for optimization problems with non-differentiable objective functions, or even problems that lack a complete mathematical model, where instead a simulation has to be used. Another strength of GAs is their ability to handle complex problems that have irregular fitness landscapes with many local optima, and a varying number of variables [10].

In GAs it is common to use chromosomes with lengths that are preserved during the evolutionary process. However, it is also possible to use a so called non-homologous crossover operator that allows the chromosomes to vary in length during the evolution [12]. Such a crossover operator was implemented in this study, see Sect. III-C for details.

In the simulations, the ego vehicle is controlled by the evolved driver model, whereas the 9 surrounding vehicles are controlled by the standard IDM described in Sect. II-A. The driver model of the ego vehicle consists of a sequence of rules and actions (described below) which are executed with a sampling interval of $\Delta t = 0.1$ s.

A. Chromosome encoding

In the encoding used, a chromosome encodes a set of *instructions*, each represented by 4 genes, $g_1, \ldots, g_4$. The first two numbers in each instruction are integers and represent which rule or action that should be used, and which relative lane that should be considered. The last two parameters are floating-point numbers that have different meanings for different instructions. The detailed interpretation of the numbers is described in Table III. For example, the instruction $[0, 1, 0.2, 0.7]$ would be translated to the instruction (rule): *If there is a vehicle in the left lane, in the interval -60 m to 40 m longitudinally, relative to the ego vehicle, then* ... A braking or accelerating pedal level is linearly mapped to the allowed negative or positive acceleration ranges mentioned in Sect. II-C.

A chromosome is constructed from a variable number of instructions such that, when decoded, it generates a driver model in the form of a list of *rule-action units*. For example, a case with three rule-actions units, could take the form:

- *Rule 1*
  - *Rule 2*
    - ▷ *Action 1*
- *Rule 3*
  - ▷ *Action 2*

▷ *Action 3*

Note that every rule-action unit contains precisely one action, and any number of rules. During evaluation of a chromosome, when deciding which action to take, the first rule is considered. If the conditions are satisfied, the next rule is considered etc. If all rules preceding an action are fulfilled, that action is performed and the evaluation is stopped. This means that at most one action can be executed. If a rule is not fulfilled, the evaluation jumps to the next group of rules preceding the next action. In the example, if, for instance, Rule 1 is not satisfied, Rule 3 would be considered next. If there are no rules associated with an action, as in Action 3 in the example, this action will be used if no preceding action has been carried out. If there is no lane available in a given direction, all rule-action units containing an action involving a lane change in that direction are ignored. Moreover, if a lane is not available, it is considered to contain no vehicles. For actions with relative lane 0 (see Table III), no lane change is performed and only acceleration is considered.

Furthermore, note that the evaluation of the chromosome always takes place every Δt s, even if the chosen action involves a lane change. In that case, the intended lane in the control model, described in Sect. II-C, is immediately switched to the desired lane, and kept there until any future action changes it.

B. Fitness measure

In order to evaluate the performance of an evolved driver model controlling the ego vehicle, the simulation environment described in Sect. II-B was used. Each scenario was initialized randomly and run until a collision occurred or the ego vehicle reached a maximum distance, d_max, in this case 500 m, solving the scenario without collision.

The fitness measure was divided into a positive and a negative part. The positive fitness, ranging from 0 to 1, was

$$f_i^+ = (d/d_\mathrm{max}) \times \min\left(\bar{v}/\bar{v}_\mathrm{ref}, 1\right), \tag{5}$$

where d is the distance driven by the ego vehicle, and d_max is the maximum distance possible. $\bar{v}$ is the average speed of the ego vehicle and $\bar{v}_\mathrm{ref}$ is the average speed when applying the reference model presented in Sect. II-A. As can be seen in the equation, the speed part was limited to the range 0 to

TABLE III
ENCODING OF AN INSTRUCTION, CONSISTING OF 4 GENES

Gene	Value	Interpretation
g_1	0	Rule: If there is a vehicle in lane g_2, in the interval g_3 to g_4 longitudinally, relative to the ego vehicle
	1	Rule: If there is no vehicle in lane g_2, in the interval g_3 to g_4 longitudinally, relative to the ego vehicle
	2	Action: Change to relative lane g_2, brake or accelerate according to g_3, using pedal level g_4
g_2	-1	Right lane
	0	Current lane
	1	Left lane
g_3	$v_3 \in [0, 1]$	if g_1 = 0 or 1, map value v_3 to [-100,100] m if g_1 = 2, g_3 represents braking if $v_3 < 0.5$ and acceleration if $v_3 \geq 0.5$
g_4	$v_4 \in [0, 1]$	if g_1 = 0 or 1, map value v_4 to [-100,100] m if g_1 = 2, g_4 represents pedal level

1, which means that no additional fitness score was given to individuals driving faster than the reference model. This was so, since the model is intended to emphasize safety.

Therefore, when a collision occurred, the distance driven by the ego vehicle was the dominant limiting factor. By contrast, if a model managed to drive without collision, the fitness was determined by the average speed.

Regularization was implemented, in order to avoid overfitting, as a negative fitness contribution depending on the length of the chromosome. Here, up to n instructions were allowed without penalty, but beyond that limit, a negative fitness contribution was computed, somewhat arbitrarily, as

$$f^- = \beta \max\left(0, L/4 - n\right), \tag{6}$$

where L is the length of the chromosome. Since there are 4 genes per instruction, $L/4$ is the number of instructions. Based on preliminary investigations, suitable values for n and β were found to be 20 and 0.2, respectively.

In every GA run, each driver model was evaluated using $i_0 = 10$ random scenarios. Then, whenever all scenarios were solved without collision and the speed part of the fitness was greater than a given threshold (set to 0.85) in 95% of the considered scenarios, an additional scenario was added. The total fitness f was then calculated as the sum of the positive fitness for every scenario minus the overall negative fitness based on the chromosome length, as

$$f = \sum_i f_i^+ - f^-. \tag{7}$$

C. Evolutionary operators and parameters

In each GA run, a population of m individuals was randomly initialized with between 10 and 20 instructions each. The individuals were encoded according to Sect. III-A and were decoded and evaluated as described in Sect. III-B.

When a new generation had been evaluated, a standard tournament selection procedure was applied, with tournament size k and tournament selection parameter p_t. Then crossover was applied with probability p_c to every pair of selected individuals. A non-homologous two-point crossover operator was used, in which the section between the two crossover points was swapped between the two selected individuals, thus allowing the length of the chromosomes of the new individuals to be different from their parents. The crossover points were chosen between the instructions.

TABLE IV
PARAMETERS USED FOR THE EVOLUTIONARY ALGORITHM.

Population size, m	40
Tournament size, k	4
Tournament selection parameter, p_t	0.8
Crossover probability, p_c	0.8
Mutation rate	$1/L$

Following selection and crossover, the individuals were exposed to random mutation. Mutations occurred with a probability computed as the inverse of the chromosome length L, i.e. on average one gene per chromosome was mutated. Note that this means that by mutating the first gene of an instruction, a rule can change into an action and vice versa. In addition to the parametric mutation just described, two additional mutation types were used (also with probability $1/L$). The first one either inserted or deleted one instruction, i.e. one rule or one action. The second one inserted or deleted a rule-action unit. In case of addition, the rule-action unit was taken as a mutated copy of one of the already existing rule-action units in the chromosome.

Finally, elitism was used, meaning that a single copy of the best individual in a given generation was inserted in the next generation without modification. The numerical values of the parameters are summarized in Table IV.

IV. RESULTS

Three different runs (Runs 1 - 3) were carried out with different random seeds of the GA. The resulting fitness variation over the generations is shown in Fig. 3, showing an early increase in fitness as more and more scenarios are solved. However, in all three runs, when eventually a general solution is found, which solves all 500 cases without collision at which point, the fitness jumps to just below 500 (since the average speed of the ego vehicle is generally smaller than that of the reference model in a few scenarios). The final driver model from one of these runs (Run 1) is shown in Table V. The evolved driver model generates a behavior where the vehicle stays in its lane and accelerates if no vehicle is close in front of it. If a vehicle is present, but there is no vehicle in the left lane, it changes lanes to the left lane. If also the right lane is occupied, it stays in its own lane and brakes. Otherwise it changes to the right lane.

It should be noted that some of the rules or rule-action units do not have any effect on the behavior of the generated driver model, i.e. they are never executed (see e.g. the second and third rule in the fourth rule-action unit, and rule-action unit 7 in Table V). For a discussion on why they are kept, see Sect. V. Furthermore, note that rule-action unit 7 and 8 can only be considered when the vehicle is positioned in the right lane, resulting in rule-action unit 6 being ignored.

In order to investigate whether or not an even simpler driver model could be found, an additional run (Run 4) was carried out, with negative fitness contribution applied from the beginning (i.e. $n = 0$ in Eq. 6). For this run, the optimization was initialized with a population consisting of mutated copies of the the final best individual of Run 1. Furthermore, this time, since a feasible structure for solving the problem was already present in the population, the crossover operator was removed. Evolution was therefore only done by selection and

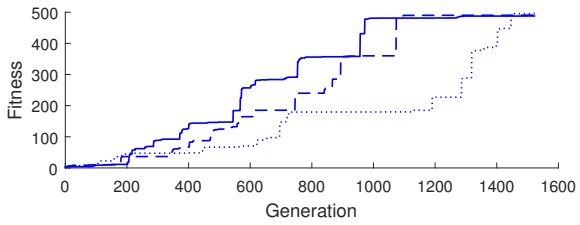

Fig. 3. Fitness variation of the best individual in the population, for three optimization runs (Runs 1 - 3) with different random seeds.

TABLE V
THE RESULTING DRIVER MODEL FROM RUN 1.

- If no vehicle in ego lane is within [-2.0, 21.5] m
  - If no vehicle in right lane is within [28.5, 62.0] m
    - ▷ Keep lane, accelerate with pedal level 0.94
- If no vehicle in left lane is within [-19.5, 80.5] m
  - ▷ Do lane change to the left, accelerate with pedal level 0.65
- If no vehicle in left lane is within [-18.6, 82.3] m
  - ▷ Do lane change to the left, brake with pedal level 0.12
- If vehicle in right lane is within [-92.0, 90.1] m
  - If no vehicle in ego lane is within [-2.2, 37.1] m
    - If no vehicle in ego lane is within [-2.2, 37.1] m
      - ▷ Keep lane, accelerate with pedal level 0.94
- If vehicle in right lane is within [-67.6, 91.3] m
  - If vehicle in right lane is within [-18.1, 81,7] m
    - ▷ Keep lane, brake with pedal level 0.88
- ▷ Do lane change to the right, accelerate with pedal level 0.80
- If no vehicle in ego lane is within [-44.5,85.3] m
  - ▷ Keep lane, brake with pedal level 0.88
- If vehicle in left lane is within [-77.1,46.6] m
  - If no vehicle in ego lane is within [28.6,84.2] m
    - ▷ Keep lane, brake with pedal level 0.88

TABLE VI
THE RESULTING DRIVER MODEL FROM RUN 4, WHICH PENALIZED THE TOTAL NUMBER OF INSTRUCTIONS.

- If no vehicle in ego lane is within [-3.4, 21.5] m
  - If no vehicle in right lane is within [63.3, 99.7] m
    - ▷ Keep lane, accelerate with pedal level 0.94
- If no vehicle in left lane is within [-17.8, 77.2] m
  - ▷ Do lane change to the left, accelerate with pedal level 0.97
- If no vehicle in ego lane is within [-2.2, 38.1] m
  - ▷ Keep lane, accelerate with pedal level 1.00
- If vehicle in right lane is within [-18.1, 40.9] m
  - ▷ Keep lane, brake with pedal level 0.88
- ▷ Do lane change to the right, accelerate with pedal level 0.78
- If vehicle in left lane is within [-75.7, 46.6] m
  - ▷ Keep lane, brake with pedal level 0.88
- ▷ Do lane change to the left, accelerate with pedal level 0.86

mutation. Over 300 generations, the overall fitness improved from 483 to 491. This was partly due to a shortening of the best chromosome, reducing the negative part of the fitness measure, and partly due to further optimization of the parameters of the instructions. The final driver model is shown in Table VI.

The resulting driver model from Run 4 was finally applied to 500 new scenarios, different from the ones over which it was trained. Importantly, the model ran through all these scenarios without collisions. Fig. 4 shows a histogram of the speed ratio $\gamma = \bar{v}/\bar{v}_{\mathrm{ref}}$ over the 500 test scenarios. This figure shows that, even though the average speed of the evolved model generally was close to that of the reference model, there are also some outliers, which are either better or worse (in terms of average speed) than the reference model. This is due to the randomness of the scenarios, where even a reasonable action can lead to a situation where the ego vehicle gets locked in, without a possibility to overtake the surrounding vehicles. Since the

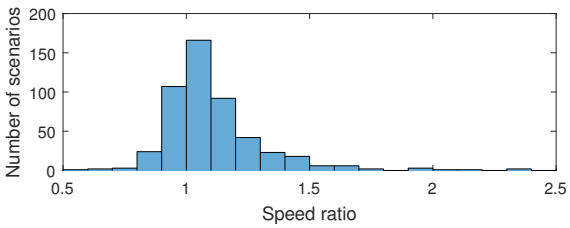

Fig. 4. Histogram of the speed ratio $\gamma = \bar{v}/\bar{v}_{\mathrm{ref}}$, obtained when applying the resulting driver model from Run 4 to 500 new scenarios.

models behave differently, such situations can occur both for the evolved model and the reference model, which explains the outliers. Note that, even though no additional fitness was given for models driving faster than the reference model, the average of γ was 1.11, indicating that the evolved model is slightly more efficient than the reference model.

V. DISCUSSION

The results show that useful driver models, whose performance equals or surpasses that of the reference model in terms of safety (at least over the scenarios considered; see below) and average speed, can be obtained with the proposed method. Reassuringly, the behavior shown by the driver models presented in Tables V and VI is similar to a gap acceptance model for lane changes, as presented in e.g. [3] or [4], where a lane change can be performed if the gap in the traffic is large enough. Moreover, unlike most driver models, the approach presented here is not limited only to a single, specific case (e.g. one-way traffic). In order to illustrate this property, a similar traffic simulation as that described in Sect. II-B was set up, with two lanes but with *oncoming traffic* in the left lane, see Fig. 5. The ego vehicle started in the right lane, with another vehicle in front, following a slow speed profile. Two oncoming vehicles were randomly placed between 200 and 1000 m ahead of the ego vehicle in the oncoming lane. They both followed a fast speed profile. The same evolutionary algorithm as described in Sect. III was applied to this case, using the same type of instructions as before (see Table III). Here, the fitness measure was simply taken as $f_i^+ = (d/d_{\max}) \times (\bar{v}/\bar{v}_{\mathrm{front}})$, where $\bar{v}_{\mathrm{front}}$ was the mean speed of the slow vehicle driving ahead of the ego vehicle. This measure was chosen since, here, there was a clear motivation for the ego vehicle to overtake the slow-moving vehicle in front. With this measure, a driver model that simply follows the slow vehicle will achieve a fitness of (just) below one, whereas if the slow vehicle is overtaken, the fitness value will be larger than 1 (if collisions are avoided, of course). The variation of the fitness from three GA runs for this case is shown in Fig. 6. As in the previous case, all 500 scenarios were solved without collision. Also, no collisions occurred in a re-evaluation involving 500 new scenarios (not used during training). In all cases, the ego vehicle overtook the slower vehicle, resulting in fitness values above 1. This shows that the proposed method is able to solve other cases, with a slight modification of the simulation environment and the fitness measure.

As mentioned in Sect. IV, some duplicate rules and unreachable actions that cannot affect the result, see Table V, were typically present in the obtained models. These non-effective instructions, referred to as introns, often have positive

Fig. 5. Example of a traffic situation for an overtaking scenario with oncoming traffic, showing the situation after 10 seconds of driving. The ego vehicle (a truck-trailer combination) is shown in red, in the bottom lane. The arrows represent the velocities of the vehicles.

effect on the progress of the evolutionary search progress, see e.g. Ch. 7 in [13] for more details. Therefore, introns should not be removed from the chromosomes during the evolution, but instead, if desired, *after* that stage, when the generated final individual will be operating the vehicle.

As described in Sect. IV, the training was carried out in two stages: An initial stage (exemplified by Runs 1-3) aimed at finding a successful driver model, and a second stage (exemplified by Run 4) aimed at simplifying the model. Attempts were also made to carry out the entire training in a single run, with a length penalty already from the first instruction, but in those cases the optimization algorithm typically got stuck in a local optimum, and the corresponding driver model was then unable to complete all scenarios. Thus, the proposed two-stage optimization procedure brought clear benefits.

It is important to note that this method only solves the type of situations to which it is exposed in the simulations. Therefore, it is crucial to set up the simulations correctly, to ensure that they cover the intended case. Furthermore, as with many machine learning methods, it is hard to guarantee functional safety with a learned driving model. One way to deal with this, commonly proposed in literature, see e.g. [14], is to use an underlying safety layer, which verifies that a planned trajectory is safe before forwarding it to the vehicle control system.

For the highway case a test was made in which more complex rules were added, involving the speed of the surrounding vehicles. These would for example have the form *If the relative speed of the vehicle in the left lane is within the range 0 to 5 m/s, then* ... However, adding such rules did not improve the performance. On the contrary, with these rule types added, the optimization quickly got stuck in a local optimum, unable to handle more than a few scenarios.

Thus, in the end, in all cases considered here, the instruction encoding shown in Table III was used. This instruction set would probably be applicable in some other cases as well. However, a more general instruction set would be needed for some cases, e.g. cases involving crossings. One must then be careful to avoid the problems just described, where the optimization gets stuck in a local optimum. Some more techniques to avoid overfitting could possibly help in this regard. Defining more general instructions is, however, a topic for future work. If additional instructions, including more action types, are introduced, allowing more than one action to be executed at every time interval could be considered. This would also make it interesting to study alternative types of chromosome representations. Moreover, investigating further the effects of different choices of the fitness and evolutionary parameters, presented in Sect. III, is also a topic for future work.

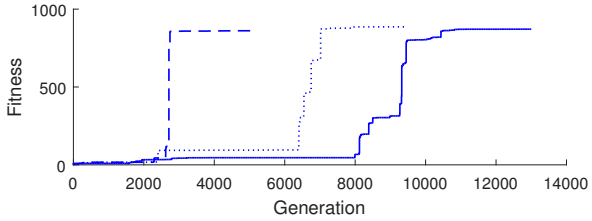

Fig. 6. Fitness variation of the best individual in the population, for three optimization runs with different random seeds.

VI. Conclusion

In conclusion, the main result of this paper is that the proposed method, involving a genetic algorithm with length-varying chromosomes, can automatically produce successful driver models in the form of a sequence of rules and actions. Here, a highway driving case was considered, with a truck-trailer combination as the ego vehicle. The resulting driver models match or surpass the performance of a reference model based on the IDM and the MOBIL model in terms of average speed. Furthermore the generality of the model has been demonstrated by applying it to a second case, in which oncoming traffic was considered. In both cases, the evolved driver model handled all scenarios without collision. Important topics for future work is to extend the model to include even more general instructions, and also to consider additional safety aspects.

Acknowledgment

The authors would like to thank Adj. Prof. Leo Laine for valuable comments on the manuscript. This work was partially supported by the Wallenberg Autonomous Systems and Software Program (WASP), and Vinnova FFI.

References

[1] K. Aghabayk *et al.*, "A state-of-the-art review of car-following models with particular considerations of heavy vehicles," *Transport Reviews*, vol. 35, no. 1, pp. 82–105, 2015.
[2] M. Treiber *et al.*, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, pp. 1805–1824, 2000.
[3] P. Gipps, "A model for the structure of lane-changing decisions," *Transportation Research Part B: Methodological*, vol. 20, no. 5, pp. 403 – 414, 1986.
[4] K. I. Ahmed, "Modeling drivers' acceleration and lane changing behavior," Ph.D. dissertation, Massachusetts Institute of Technology, 1999.
[5] A. Kesting *et al.*, "General lane-changing model mobil for car-following models," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1999, pp. 86–94, 2007.
[6] J. Eggert and F. Damerow, "Complex lane change behavior in the foresighted driver model," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sept. 2015, pp. 1747–1754.
[7] S. Ulbrich and M. Maurer, "Towards tactical lane change behavior planning for automated vehicles," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, pp. 989–995.
[8] O. Benderius, "Modelling driver steering and neuromuscular behaviour," Ph.D. dissertation, Chalmers University of Technology, 2014.
[9] P. Nilsson *et al.*, "A driver model using optic information for longitudinal and lateral control of a long vehicle combination," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 1456–1461.
[10] J. H. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.
[11] D. D. Salvucci and R. Gray, "A two-point visual control model of steering," *Perception*, vol. 33, no. 10, pp. 1233–1248, 2004.
[12] M. F. Brameier and W. Banzhaf, *Linear genetic programming*. Springer Science & Business Media, 2007.
[13] W. Banzhaf *et al.*, *Genetic Programming: An Introduction: on the Automatic Evolution of Computer Programs and Its Applications*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998.
[14] S. Underwood *et al.*, *Truck Automation: Testing and Trusting the Virtual Driver*. Springer International Publishing, 2016, pp. 91–109.