A data-driven behavior generation algorithm in car-following scenarios

Yihuan Zhang & Jun Wang
Department of Control Science and Engineering, Tongji University, Shanghai, P. R. China

Qin Lin & Sicco Verwer
Department of Intelligent Systems, Delft University of Technology, Delft, the Netherlands

John M. Dolan
The Robotic Institute, Carnegie Mellon University, Pittsburgh, PA, United States

ABSTRACT: The conventional Adaptive Cruise Control system lacks user-friendly design. In this paper, a novel method for learning a generative model from human drivers' car-following data using an automaton learning algorithms is proposed. By partitioning the model using frequent common state sequences, human driving patterns are extracted and clustered. Then a cluster identification method is used to obtain the current driving pattern and generate a desired acceleration. The experiments validate that the simulated trajectories of the proposed method are more similarly to human drivers than those of conventional PID controller.

1 INTRODUCTION

An Adaptive Cruise Control (ACC) system assists drivers to maintain safety spacing from lead vehicles and eases drivers' tedious workload of frequent acceleration and deceleration operations. It has been well studied and is regarded as a cornerstone towards autonomous driving. However, the drawbacks of an ACC system are evident: 1) driver's overconfidence or distrust in the system; 2) a mode awareness error when the system consists of two types of ACCs e.g., a high-speed-range ACC and a low-speed-range ACC; 3) a difference in timing of acceleration/deceleration between drivers and system by Hiraoka, 2005. These are essentially caused by the inconsistency between systems and human drivers, since the control algorithm of an ACC focuses more on mathematical optimization of safety and comfort rather than driving behaviors.

Unlike a conventional ACC system without user-friendly design, our work aims at learning real human drivers' car-following behaviors (from the Next Generation SIMulation (NGSIM) dataset) using a state-of-the-art real-time automaton learning algorithm. We use such a generative model as a simulator/generator to mimic human drivers' car-following behaviors. A car-following model essentially reflects how a driver responds to his or her existing driving state by implementing a certain action. A more formal definition is that this model tries to bridge (linearly by Pipes, 1953 and Helly, 1900 or non-linearly by Gazis, 1961 and Treiber, 2000) input stimuli or explanatory variables, like subject vehicle speed, relative distance and speed to a lead vehicle, and output actions or response variables, like acceleration. Genetic algorithms are the most widely used techniques to identify parameter values in car-following models. A gross fitting strategy, i.e., fitting a car-following model on all the collected data, is usually used for identification (sometimes using two models for congested and uncongested traffic situations proposed by Ceder, 1976). Gross fitting has inevitably large fitting errors and it is more suitable for use in overall traffic flow simulating. Heterogeneity of driving behavior including inter-driver difference proposed by Hoogendoorn, 2006 (different car-following models may apply to different drivers, also possible for driving skills evaluation of individual drivers) and intra-driver difference by Hamdar 2008 (individual drivers may change their behavior over the data collection period) has been reported in the literature by Hinsbergen, 2015. In this paper, our approach

is similar to the idea of intra-driver difference but lies on a mesoscale to model driving behaviors/patterns shared by drivers.

We deploy a latent state sequence clustering (cooperating the learned timed automaton) to input stimulus (speed, relative distance, relative speed). Afterwards, we learn a car-following model in each individual cluster representing a driving pattern. We claim that a complete car-following model consists of such driving patterns. Using such a divide-and-rule or piece-wise fitting, the approximation error of this switching car-following model is expected to be lower. Furthermore, the experiments demonstrate valid and human-like car-following trajectories. This approach sheds light on a novel driver-specified ACC system design from real car-following data.

2 BEHAVIOR GENERATION ALGORITHM

Fig. 1 shows a flowchart to provide an overview of our approach. First, a PDRTA (probabilistic deterministic real timed automaton) is learned from the NGSIM dataset. Verwer, 2011 proposed a PDRTA which is essentially a timed variant of a PDFA (probabilistic deterministic finite automaton, similar to a Hidden Markov Model). Then frequent common substrings in (latent) state sequences are extracted and a hierarchical clustering method is applied to obtain the clusters. Each cluster contains a couple of states representing a driving pattern, e.g., short, medium, and long distance car-following. We calibrate a car-following model in each individual cluster. The current cluster of the subject vehicle is determined by its current state and then the corresponding car-following model is selected to generate the desired acceleration. The status of the subject vehicle (velocity, relative velocity, and relative distance) is continuously updated online using the acceleration computed in the last time step.

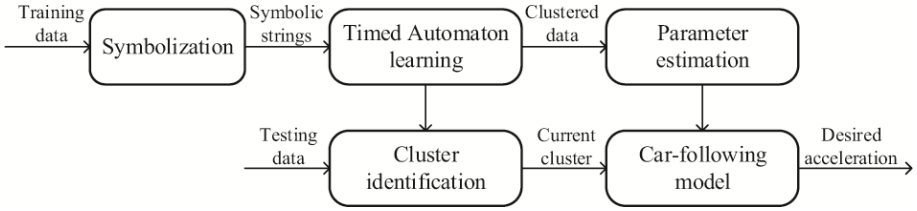


Figure 1. Flowchart of the proposed approach.

2.1 *Car-following model identification*

Traditional car-following model identification, also called calibration in many papers, is that given an assumed model we are trying to identify its parameters. In this paper, the linear car-following model (Helly model) is used. The acceleration in Helly's car-following model is a linear function combining the relative speed and the relative distance between the headway and the desired headway, which is defined by Helly, 1900:

$$a(t) = C_1 \cdot \Delta v(t-\tau) + C_2 \left(\Delta x(t-\tau) - D(t)\right) \tag{1}$$

$$D(t) = \alpha + \beta \cdot v(t-\tau) + \gamma \cdot a(t-\tau) \tag{2}$$

where C_1, C_2, α, β, γ and τ are constant parameters that need to be calibrated. The desired headway is a function of the velocity and the acceleration of the follower vehicle, where α, β and γ are the corresponding parameters for those variables. Also, τ represents the reaction time delay of the follower vehicle. In this paper, the differential evolution algorithm (DEA) proposed by Storn, 1997 is applied to identify the parameters of the Helly car-following model.

2.2 *Data description and Symbolization*

The public dataset on individual vehicle trajectories we use in this paper is from NGSIM, 2007, a program funded by the U.S. Federal Highway Administration. This trajectory data are so far unique in the history of traffic research and provides a great and valuable basis for validation and calibration of microscopic traffic models by Thiemann, 2008. We test our method in the datasets from the I80 highways.

The I80 dataset consists three 15-minute periods: 4:00 p.m. to 4:15 p.m., 5:00 p.m. to 5:15 p.m., and 5:15 p.m. to 5:30 p.m. These periods represent the buildup of congestion, or the transition between uncongested and congested conditions, and full congestion during the peak period. In both the I80 dataset, vehicle trajectory data provide precise location of each vehicle within the study area every one-tenth of a second. Based on the trajectory data, following and leading vehicle pairs are extracted for car-following behavior studying. Table 1 shows a summary of data features used in our paper. Note that vehicle speed, relative distance, and relative speed are explanatory variables as inputs. Longitudinal acceleration is a response variable as output and used as a ground truth for testing the model's output.

Table 1. Extracted features from NGSIM data.

Feature	Definition
Vehicle speed	Speed of a subject vehicle
Longitudinal Acceleration	Acceleration of a subject vehicle
Relative distance	Distance from the front of a subject vehicle to the back of a lead vehicle
Relative speed	Speed difference between a subject vehicle and a lead vehicle

The *k*-means clustering algorithm is used as a discretization approach to symbolize the car-following data. The ELBOW method proposed by Goutte, 1999 is used to determine the optimal number of clusters. The idea is to find the number of clusters that stops sharp dropping of the WSS (within the cluster sum of squares). In this paper, we choose 10 as the number of symbols. Symbolic strings are then converted to timed strings using all 3 input features at once using the method proposed by Zhang, 2017.

2.3 *PDRTA learning*

A state-of-the-art machine learning algorithm named RTI+ is used to learn car-following behaviors from unlabeled data. For more details about this algorithm, readers are referred to the thesis of Verwer, 2010a. Traditional probabilistic state merging algorithm starts by building a large tree-shaped automaton called prefix tree from a sample of input strings. Every state of this tree can be reached by exactly one untimed string and therefore encodes exactly the input sample. The algorithm then greedily merges pairs of states *(q, q')* in this tree, forming a smaller and smaller machine that generalizes over samples. Because PDRTAs are deterministic, for every event the states that are reached from q and q' have to be merged as well (the determinization process). The algorithm uses a statistical test to decide whether to merge or not. A merge between state pair q and q' is considered good if the future behavior after reaching q is similar to that after reaching q', which can be tested using, e.g., a likelihood-ratio test proposed by Verwer, 2010b. This essentially tests the Markov property, i.e., whether future behavior is independent of being in state q or q'. When these futures are significantly different, the merge is considered inconsistent and will not be performed. A high-level overview of RTI+ is in Algorithm 1. A hierarchical clustering is deployed on latent states to find the similar pattern in the car-following model. Readers are referred to our previous work by Zhang, 2017 for the detailed implementation. The final model is partitioned into several regimes (see blocks with different colors in Fig. 2). The DEA algorithm is used to estimate the parameters of the Helly model in each cluster.

Algorithm 1. Data identification with RTI+.

Input: A (multi-)set of timed strings S_+
Output: A small PDRTA for S_+

Construct a timed prefix tree from S_+, let $Q'=\emptyset$;
for all transitions from S_+, **do**
  Evaluate all possible merges of q' with states from Q';

```
    Evaluate all possible splits of transitions;
    if the lowest split p-value < 0.05 then
        Perform this split;
    end
    else if the highest merge p-value > 0.05 then
        Perform this merge;
    end
    else
        Add q to Q';
    end
end
```

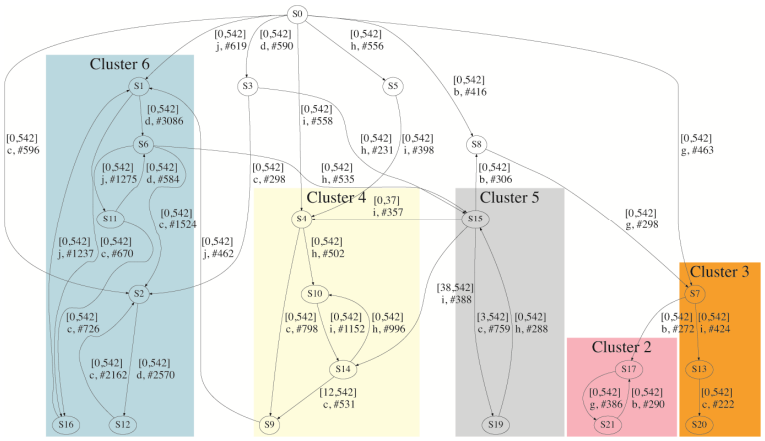


Figure 2. Illustration of the final timed automaton.

2.4 *Cluster identification*

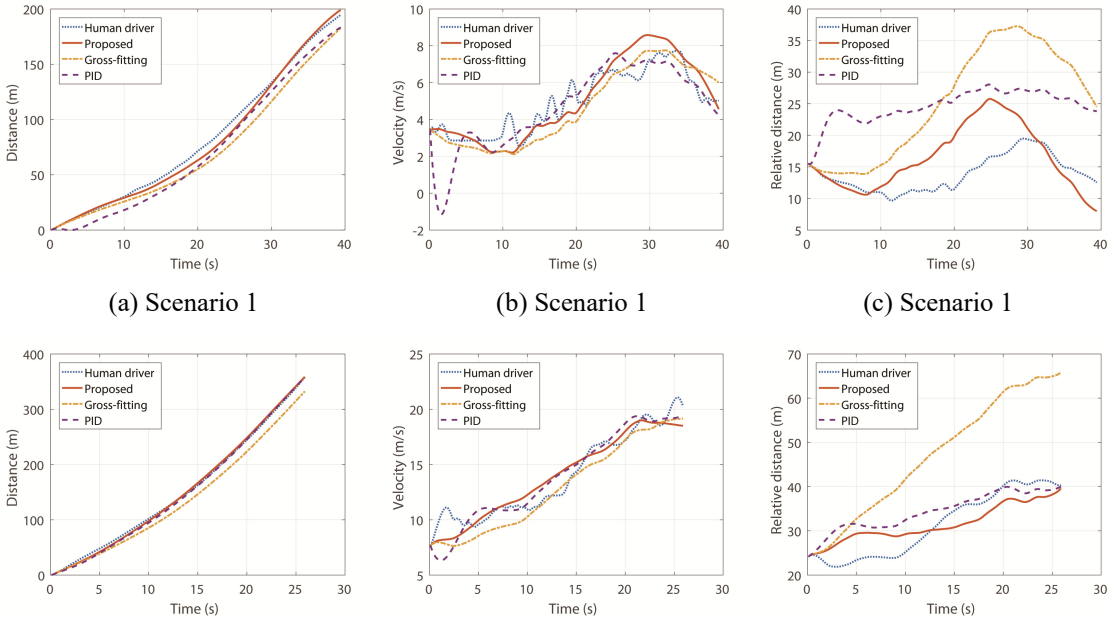
The behavior generating method is detailed in this part. First, the current status of the leading vehicle and the subject vehicle, i.e., the velocity, the relative distance and relative velocity, are symbolized according to the *k*-means code book. The corresponding state in the timed automaton is derived by the symbolic transition. The state is afterwards assigned a cluster's ID (blocks of driving patterns in Fig. 2). Finally the corresponding car-following model from such a cluster is used to generate the acceleration as output of the subject vehicle. The subject vehicle's movement is updated iteratively using the output acceleration. The updated driving status of the leading vehicle and the subject vehicle will be circularly used to trigger the state transition in the timed automaton. Note that the state in the timed automaton transits only if necessary by a valid external timed event's trigger. For example, the current state *S15* transits to *S14* when the input event is *i* while the time difference from *i* to its precedent event is validly inside the time guard [38, 542].

A good simulated car-following trajectory should intuitively be close to the real one in the dataset. A standard PID-based ACC controller is designed. Gross-fitting model is another baseline learning a single model without clustering. Fig. 3 shows that our model controls the subject vehicle to follow the lead vehicle very well. Although it is not very sensitive to frequent change of the lead vehicle's velocity (also the case in other models), it follows the overall trend. Kesting, 2008 proposed the relative error F_{rel}, the absolute error F_{abs} and the mixed error measure F_{mix} as indicators to quantitatively evaluate trajectories, which are defined as follows:

$$F_{rel} = \sqrt{\left\langle \left(\frac{s^{sim} - s^{data}}{s^{data}} \right)^2 \right\rangle}, F_{abs} = \sqrt{\frac{\left\langle \left(s^{sim} - s^{data} \right)^2 \right\rangle}{\left\langle s^{data} \right\rangle^2}}, F_{mix} = \sqrt{\frac{1}{\left\langle \left| s^{data} \right| \right\rangle} \left\langle \frac{\left(s^{sim} - s^{data} \right)^2}{\left| s^{data} \right|} \right\rangle} \tag{3-5}$$

where $\langle \bullet \rangle$ is the average value of the time sequence, while s^{sim} and s^{data} are the simulated and human drivers' data, respectively.

We learn our proposed model and gross-fitting model from the training data (80% of the whole trajectories in the dataset) and evaluate in the testing data (20% of the whole dataset). The results are reported in Table 2, which shows our model outperforms two other baselines.



(a) Scenario 1 (b) Scenario 1 (c) Scenario 1

(d) Scenario 2 (e) Scenario 2 (f) Scenario 2

Figure 3. Trajectory simulation in two example scenarios: (a) - (c) and (d) - (f). The longitudinal distance, velocity and relative velocity are plotted in comparison of three methods: the proposed method, the gross-fitting method and the PID method.

Table 2. Performance comparison.

		Proposed generator	Gross-fitting generator	PID controller
F_{rel}	Mean ± Std.	**0.1157 ± 0.0807**	0.1332 ± 0.0796	0.2466 ± 0.2852
F_{abs}	Mean ± Std.	**0.0764 ± 0.0643**	0.1091 ± 0.0850	0.1105 ± 0.0875
F_{mix}	Mean ± Std.	**0.0766 ± 0.0615**	0.1034 ± 0.0781	0.1360 ± 0.0973

3 CONCLUSION AND FUTURE WORK

In this paper, we propose a framework for a human-like car-following controller design. The experiments validate that the simulated trajectories of our approach are more similar to human drivers than those of conventional PID controller. A specified controller for an individual driver is also achievable once sufficient car-following data are available from him/her. Our model has the advantage of being an active controller, e.g., enabling a transition from a short-distance car-following status to a medium-distance one, forcing states switching in our automaton model. Comprehensive evaluations involving a standard ACC controller comparison, considering safety and stability, and combining a close loop control strategy will be studied in the future.

4 ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under Grant No. 61473209, Technologiestichting STW VENI project 13136 (MANTA), and NWO project 62001628 (LEMMA).

REFERENCES

Ceder, A., & May, A. D. 1976. Further evaluation of single-and two-regime traffic flow models. Transportation Research Record, (567).

Gazis, D. C., Herman, R., & Rothery, R. W. 1961. Nonlinear follow-the-leader models of traffic flow. Operations research, 9(4), 545-567.

Goutte, C., Toft, P., Rostrup, E., Nielsen, F. Å., & Hansen, L. K. 1999. On clustering fMRI time series. NeuroImage, 9(3), 298-310.

Helly, W. 1900. Simulation of bottlenecks in single-lane traffic flow.

Hiraoka, T., Kunimatsu, T., Nishihara, O. & Kumamoto, H. 2005. Modeling of driver following behavior based on minimum-jerk theory. Proc. 12th World Congress ITS.

Hoogendoorn, S., Ossen, S., & Schreuder, M. 2006. Empirics of multianticipative car-following behavior. Transportation Research Record: Journal of the Transportation Research Board, (1965), 112-120.

Hamdar, S., Treiber, M., Mahmassani, H., & Kesting, A. 2008. Modeling driver behavior as sequential risk-taking task. Transportation Research Record: Journal of the Transportation Research Board, (2088), 208-217.

Kesting, A. & Treiber, M. 2008. Calibrating car-following models by using trajectory data: Methodological study. Transportation Research Record: Journal of the Transportation Research Board, (2008), 148-156.

NGSIM. 2007. U.S. Department of Transportation, NGSIM - Next generation simulation, website: http://www.ngsim.fhwa.dot.gov.

Pipes, L. A. 1953. An operational analysis of traffic dynamics. Journal of applied physics, 24(3), 274-281.

Storn, R., & Price, K. 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), 341-359.

Treiber, M., Hennecke, A., & Helbing, D. 2000. Congested traffic states in empirical observations and microscopic simulations. Physical review E, 62(2), 1805.

Thiemann, C., Treiber, M., & Kesting, A. 2008. Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data. Transportation Research Record: Journal of the Transportation Research Board, (2088), 90-101.

Verwer, S. E. 2010a. Efficient identification of timed automata: Theory and practice (Doctoral dissertation, TU Delft, Delft University of Technology).

Verwer, S., de Weerdt, M., & Witteveen, C. 2010b. A likelihood-ratio test for identifying probabilistic deterministic real-time automata from positive data. In International Colloquium on Grammatical Inference (pp. 203-216). Springer Berlin Heidelberg.

Verwer, S., De Weerdt, M., & Witteveen, C. 2011. Learning driving behavior by timed syntactic pattern recognition. Proc. 22nd International Joint Conference on Artificial Intelligence pp. 1529-1534.

Van Hinsbergen, C. P. I. J., Schakel, W. J., Knoop, V. L., van Lint, J. W. C., & Hoogendoorn, S. P. 2015. A general framework for calibrating and comparing car-following models. Transportmetrica A: Transport Science, 11(5), 420-440.

Zhang, Y., Lin, Q., Wang, J., & Verwer S. 2017. Learning car-following behaviors using timed automata, in The 20th World Congress of the International Federation of Automatic Control (IFAC). IFAC, 2017.