Video-Based Personalized Traffic Learning

Qianwen Chao[a], Jingjing Shen[a], Xiaogang Jin[a,*]

[a] *State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, P.R. China*

Abstract

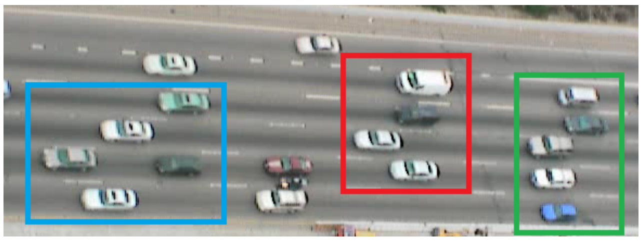
We present a video-based approach to learn the specific driving characteristics of drivers in the video for advanced traffic control. Each vehicle's specific driving characteristics are calculated with an offline learning process. Given each vehicle's initial status and the personalized parameters as input, our approach can vividly reproduce the traffic flow in the sample video with a high accuracy. The learned characteristics can also be applied to any agent-based traffic simulation systems. We then introduce a new traffic animation method that attempts to animate each vehicle with its real driving habits and show its adaptation to the surrounding traffic situation. Our results are compared to existing traffic animation methods to demonstrate the effectiveness of our presented approach.

Keywords: Traffic control, Personalized learning, Video-based, Genetic Algorithm

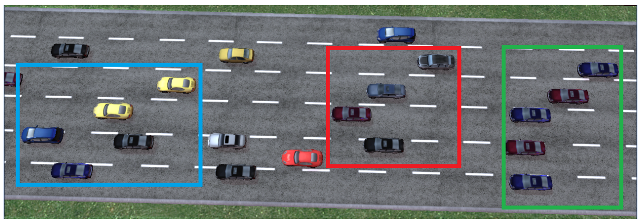
1. Introduction

With the popularity of vehicles and the dramatically increasing demand on transportation, road transport has brought about more and more serious negative effects— traffic congestion, traffic accident, and environmental pollution. Traffic management has been a global challenge with its direct impact on economy, environment, and energy. Meanwhile, traffic simulation has found its wide use in computer animation, computer game, and virtual reality [1] [2]. Some methods try to simulate each vehicle's behaviors while others aim to capture high-level flow appearance. However, the simulated results usually do not correlate to each driver's personalized driving behavior. Moreover, with better vehicle detection and tracking technology and more software tools for viewing road network, such as OpenStreetMap and GPS, there is a growing need to present realistic traffic scenarios in a virtual environment based on real-world vehicle trajectory data.

In the real world, drivers' driving behaviors vary significantly depending on time, place, personality trait, and many other social factors. These variations in driving behaviors are often characterized by observable factors such as driver's speed choice, gap acceptance, preferred rate of acceleration or deceleration, environmental adaptation factor, and so on. Estimating such characteristics is an important task if we want to reconstruct the traffic flow correlating to an input real traffic. However, existing traffic simulators set these parameters as random values around the average of empirical values, which are hard to reflect drivers' personalized driving behaviors in a specific environment. Moreover, little attention has been paid to this problem in existing traffic simulation methods.



(a)



(b)

Figure 1: One frame in the traffic sample video (a) and our reconstruction result (b).

In this paper, we propose a data-driven method to simulate virtual traffic flows that exhibit driving behaviors imitating real traffic flows. We record the motion of vehicles from an aerial view using a camcorder, extract the two-dimensional moving strategies of each vehicle in the video, and then learn the specific driving characteristics from the observed trajectories.

Learning driving behavior from videos is a challenging problem because the motion of each driver is influenced by not only the local road traffic condition, but also the driver's personality and social factors, which can not be directly seen in the captured video. We choose a short clip from the input video as the learning sample, and use a microscopic traffic model to approximate each vehicle's behavior. Since 1940s, lots of traffic simulation models have been proposed, tested and evaluated for calibra-

*Corresponding author

Email addresses: chaoqianwen@zjucadcg.cn (Qianwen Chao), shenjingjing@zjucadcg.cn (Jingjing Shen), jin@cad.zju.edu.cn (Xiaogang Jin)

tion. Among them, the intelligent driver model (IDM) [3] was proven to be one of the best approximation models and it conforms to our daily driving habits. The intelligent driver model with memory (IDMM) [4] is an expansion of IDM which introduces memory effects to describe drivers' adaption to the congested traffic. In this work, we revise the original IDMM model to describe the adaptation of drivers to the surrounding traffic situation (not limited to congested traffic).

We present a mapping between the low-level IDMM parameters and high-level driving characteristics. Inspired by the theory on the car-following model calibration [3], we utilize a nonlinear optimization scheme to compute each vehicle's optimal parameter set of IDMM. Different from previous model calibration methods, we develop an adaptive genetic algorithm to better fit for traffic animation.

The main contribution of the paper is that we introduce a novel method to estimate vehicles' personalized driving characteristics based on training data from an input video. These parameters are then employed to reconstruct the traffic flow conforming to the video. We also present a new traffic animation method using IDMM to show drivers' adaptation to the surrounding traffic situation. In addition, we propose an offline learning approach combining IDMM with an adaptive genetic algorithm, which outperforms existing methods for model calibration. Our approach can reproduce new traffic scenarios exhibiting similar driving behaviors with the sample video. Fig. 1 shows a reconstructed scenario using our method.

The rest of the paper is organized as follows. Section 2 describes the related work on traffic animation and model calibration. Section 3 presents an overview of our approach and Section 4 gives a detailed description of the algorithm. The performance analysis and simulation results are shown in Section 5. Finally, we conclude the paper and discuss the future work in Section 6.

2. RELATED WORK

In this section, we give a brief review of prior work in traffic animation and crowd behavior learning. Model calibrations and genetic algorithms are also reviewed as they will be employed in our framework.

2.1. Traffic Animation and Crowd Behavior Learning

The growing ubiquity of vehicle traffic in everyday life has attracted considerable interest in traffic behavior modeling and traffic visualization techniques. In computer graphics, much of the research on traffic has focused on two hot topics: the classical problem of traffic simulation, and traffic reconstruction [1]. The classical problem of traffic simulation is mainly about traffic behavior model. Given a road network, a proposed behavior model, and initial car states, how does the traffic evolve? In general, there are two popular classes of traffic simulation techniques: the continuum-based macroscopic and agent-based microscopic techniques.

In macroscopic simulation, traffic is treated as a kind of continuum whose evolution in time is described by partial differential equations. A famous macroscopic model was developed by Lighthill, Whitham and Richards in 1955 [5] called LWR model. It can fully describe the basic traffic-related phenomena: traffic jams and evacuation. In the 70s, Payne [6] and Whitham [7] introduced the momentum conservation equation to the original LWR model and simulated some more complicated cases using their PW model. This model was further revised by Aw, Rascle [8] and Zhang [9] to eliminate the nonphysical behavior, referred as ARZ model. In computer graphics, Sewall [10] extended the ARZ model to correctly handle lane changes, merges, and traffic behaviors due to changes in speed limit.

The agent-based microscopic methods treat each vehicle as a discrete autonomous agent with specific rules governing their behavior. Gerlough [11] summarized a set of car-following rules in his dissertation about traffic simulation in 1955. Through a variety of expansion, it has formed some new models, such as the optimal velocity model [12], the intelligent driving model (IDM) [13], and the intelligent driving model with memory [4]. In computer graphics, Shen et al. [14] proposed a new agent-based model by combining IDM with a flexible lane-changing model mainly for vivid traffic animation purpose. Sewall et al. [2] presented a hybrid traffic model integrating continuum and agent-based methods for large-scale traffic animation.

In this paper, we focus on the efficient traffic reconstruction with realistic and various driving characteristics extracted from real-world discrete spatio-temporal data. The aim is to approximate the real-world data as much as possible, and finally reproduce the real-world traffic scenarios. Compared to the classical problems of traffic simulation, this topic is less studied. Sewall et al. [1] presented a novel concept of *Virtualized Traffic*, in which traffic is reconstructed from discrete data obtained by sensors placed alongside the road. Their approach can reconstruct plausible trajectories for each car using priority-based motion planning techniques. The limitation of their approach is that they do not take the personalized behaviors of drivers into consideration.

There are some research work on data-driven behavior learning in crowd simulation. Some methods train new behavior models for crowd based on input video data. For example, Lee et al. [15] used data-driven methods to match recorded motion from videos by training a group behavior model. Ju et al. [16] proposed a data-driven method which blended existing crowd data to generate a new crowd animation. Other approaches simulate heterogeneous crowd behaviors based on perceptual data or observed personality in a crowd. For example, Guy et al. [17] derived a linear mapping between crowd simulation parameters and the personality model based on adopting results from user studies. As vehicle traffic has more strict rules than pedestrian crowds, the behavior learning methods are different.

2.2. Microscopic Car-Following Model Calibration

Microscopic car-following models provide us powerful tools to simulate the behavior for each driver, and thus they can be used to study the macroscopic traffic phenomena. The objective of model calibration is to assess the performance of car-following models using real trajectory data. The performance

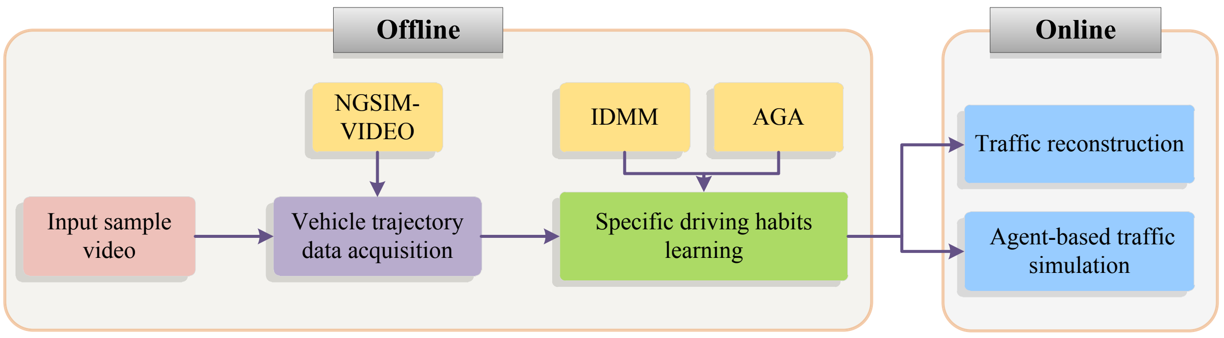

Offline
Online
NGSIM-VIDEO
IDMM
AGA
Input sample video
Vehicle trajectory data acquisition
Specific driving habits learning
Traffic reconstruction
Agent-based traffic simulation

Figure 2: The framework of our system. Trajectory data acquisition and driving habits learning procedures are both implemented in an offline module.

of car-following models greatly relies on the parameter set they use to describe and control the vehicle's motion.

In the model calibration process, the car-following model parameters need to be adjusted until an acceptable match is found between the simulated model dynamics and the observed drivers' behavior. Engineering judgment and trial-and-error methods are still widely used especially in the industry [18]. More systematic approaches, including the gradient method [19] and the genetic algorithm [20], address the model calibration procedure as an optimization problem: a combination of parameter values is searched that an objective function (error term) is minimized. Kesting and Triber calibrated and compared IDM and Optimal Velocity model with Bosch GmbH dataset using the genetic algorithm method ([3], [21]). The result shows that IDM is an acceptable approximation to our daily driving habits, and usually has smaller tracking gap error than other models.

There exists a vast amount of literature on calibrating models in the field of traffic. However, in computer graphics, no prior work exists in utilizing model calibration to estimate personalized driving behaviors that are critical for detailed realistic traffic animations. Our approach focuses on estimating the optimal parameter set for each vehicle to produce high-quality traffic reconstruction and realistic simulation.

2.3. *Genetic Algorithm (GA)*

Genetic algorithms (GAs) are robust search and optimization techniques based on principles of evolution and natural selection. These algorithms convert the problem in a specific domain into a model using a chromosome-like data structure. They evolve the chromosomes using selection, recombination, and mutation operators. The great benefits of GAs are that they are efficient, concurrent, and can adaptively control the search process to reach the globally optimal solution. In fact, GAs were successfully used in many aspects of traffic field, such as optimal traffic signal control [22], traffic assignment [23], and traffic model calibration ([20], [3], [21]). In traffic model calibration, genetic algorithms are superior to the gradient techniques as the search is not biased towards the locally optimal solution. More details on GA can be found in [24], [25].

3. Method

In this section, we briefly present an overview of our approach and discuss the microscopic car-following model we have employed for traffic simulation.

3.1. *Overview of methodology*

Our system aims to learn and apply drivers' individual driving characteristics from the video sample. The system framework is shown in Fig. 2. The main process can be divided into three phases: the acquisition of each vehicle's trajectory data, the learning of each vehicle's unique driving habits, and the online traffic animation (reconstruction or simulation). The first two phases are both manipulated in an offline module.

Since we are interested in learning each driver's driving characteristics in the input video, data acquisition from video samples is not the focus of this paper. We simply use the NGSIM-VIDEO software [26] to preprocess video images, and then extract the vehicle trajectories in each frame. More details about NGSIM-VIDEO can be found in [26].

Learning drivers'specific driving characteristics from videos is a challenging problem because each vehicle's behavior is influenced by both the driver's preferences and local road traffic situations. In this work, we assume that each vehicle's motion is controlled by its driver's personality, features of the environment, and the motion of nearby individuals. We use the intelligent driver model with memory (IDMM) as the underlying behavior control model to describe the decision making process of each individual vehicle. Accordingly, we formulate the estimation of each vehicle's unique driving habits as a problem to find its specific optimal parameter set of a microscopic driving model. In the following, we will describe the basic simulation techniques of IDMM.

3.2. *Intelligent Driver Model with Memory (IDMM)*

IDMM is a microscopic car-following model extended from IDM [13]. We first give a short introduction to IDM. The traffic state at a given time in IDM is characterized by the positions, velocities, and the lane indexes of all vehicles. The driver's decision of accelerating or braking depends on its current driving speed, the relative speed and position to its leader (that is, the vehicle in front of it on the same lane). The relationship

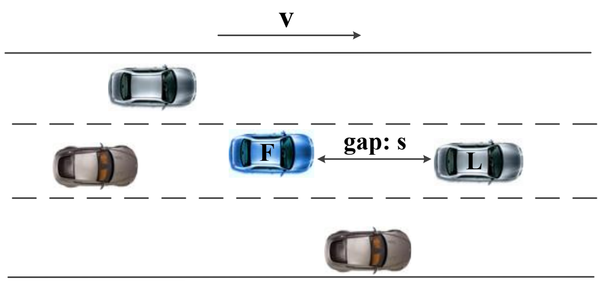


Figure 3: The relationship between the involved vehicles: the Leader (L), the Follower (F) and their gap (s).

between the involved vehicles is shown in Fig. 3. The following driver is modeled to take actions as response towards the surrounding traffic condition, for purpose of maintaining a safe gap to the leading vehicle while seeking for its desired velocity during driving.

IDM defines the vehicle's acceleration as a function of the vehicle's velocity v, gap distance s, and relative velocity Δv [13]:

$$a_{idm}\left(s,v,\Delta v\right)=a\left[1-\left(\frac{v}{v_0}\right)^4-\left(\frac{s^*(v,\Delta v)}{s}\right)^2\right] \tag{1}$$

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

As shown in Eq. (1), the acceleration can be divided into two parts: the first part $a_{acc}=a\left[1-\left(\frac{v}{v_0}\right)^4\right]$ indicates free acceleration towards a desired velocity v_0, while the latter part $a_{dec}=-a\left(\frac{s^*}{s}\right)^2$ represents a braking deceleration strategy according to the current gap s and the desired minimum gap to the preceding vehicle s^*. The acceleration on a free road decreases smoothly from the initial maximum acceleration a to zero when approaching the desired velocity v_0. IDM shows a stable crash-free dynamics with an intelligent braking strategy.

IDMM introduces an additional environmental adaptation factor β to the IDM to describe the adaptation of drivers to the surrounding traffic situation during the past few minutes [4]. We model this by varying the IDM acceleration in the range between a_{idm} (habitual acceleration) and βa_{idm} (adapted acceleration):

$$\begin{aligned} a_{idmm} &= \frac{v}{v_0}a_{idm}+(1-\frac{v}{v_0})\beta a_{idm} \\ &= a_{idm}[\beta+(1-\beta)\frac{v}{v_0}] \end{aligned} \tag{3}$$

Eq. (3) is an environmental adaptation term which adaptively adjusts the vehicle's acceleration according to the current velocity. $\frac{v}{v_0}$ reflects the efficiency of movement from the driver's point of view. $\frac{v}{v_0}=1$ means zero hindrance, corresponding to the acceleration a_{idm} in a free traffic; $\frac{v}{v_0}=0$ indicates maximum hindrance, corresponding to the acceleration βa_{idm} in a traffic jam. Different drivers and different traffic situation mean different values of β. In the parameter learning process, Eq. (3) can be viewed as an error correction term. We modify β value based on the current learning error and try to learn an average β value which most reflects the memory effect for this vehicle in this traffic scene. For example, when a_{idm} is smaller than its real value, β is greater than 1 which makes a_{idmm} vary in the range between a_{idm} and βa_{idm}. When a_{idm} is larger than its real value, β is less than 1, making a_{idmm} change between βa_{idm} and a_{idm}. In the special case $\beta=1$, the IDMM reverts to the original IDM. Since β is a constant for each vehicle, which reflects the overall adaption to the environment, $\frac{v}{v_0}$ dynamically adjusts the instantaneous acceleration between a_{idm} and βa_{idm} to approximate its real value.

Notice that in [4], β is coupled to the IDM parameters, such as the safety time headway T, the comfortable deceleration b, or the desired velocity v_0, using the analogous equations. In our experiments, we found that coupling β into the final a_{idm} leads to better adaptation than coupling it into a specific parameter. In addition, our IDMM can describe drivers' adaption in any traffic environment, while IDMM in [4] is mainly designed for congested traffic.

In this paper, the personalized driving behavior of each intelligent vehicle is mainly controlled by the driver's personality trait and environmental factors, which are specifically represented by the intuitive parameters (v_0,T,a,b,s_0,β) in IDMM. All these parameters have meaningful interpretations:

T is the desired safety time headway;
a is maximum acceleration;
b is comfortable braking deceleration;
s_0 is jam space headway;
v_0 is desired free-flow velocity;
β is the adaptation factor.

These parameters are usually initialized to empirical values for all vehicles, adding some random fluctuations to reflect individual diversity in locomotion. Obviously, this assignment method cannot adequately reflect the real-world traffic. In general, each "driver-vehicle unit" can be equipped with its individual parameter value settings for following reasons,

1) Different types of vehicles have diverse performance, for example, trucks are characterized by relatively low values of v_0, a, and b than cars.

2) Drivers with different personality trait own various driving habits, such as, careful drivers used to driving with a high safety time headway T, while aggressive drivers are characterized by a low T.

3) Different road conditions in different country areas set varying traffic rules, which leads to distinct values of v_0 and s_0 for the same vehicle.

4) Different drivers obviously have different adaptation to the surrounding traffic situation, which means different values of β in the IDMM.

Taking these factors into consideration, we are supposed to find each vehicle's exclusive parameter set (v_0,T,a,b,s_0,β) of IDMM in accord with the local environmental conditions and driver's personality characteristics.

4. Vehicles' Personalized Parameters Learning

As explained in Section 3, the task is to determine the optimal model parameter set $(v_0, T, a, b, s_0, \beta)$ that best fits the given data. The training data for each specific vehicle includes two parts: the first N frames of its trajectory, and the velocity and position of its leader in these frames. Each vehicle's parameters are learned independently. Inspired by the model calibration method in [3], we formulate the learning process as an optimization problem. The main differences between our approach and the model calibration are as follows: (1). The objective of our paper is to learn each vehicle's personalized driving parameters for realistic animation, while the model calibration mainly focuses on evaluating the accuracy of the model. (2). We effectively improve the simple genetic algorithm (SGA) in [3] and propose a new adaptive genetic algorithm (AGA) which shows better performance than the existing methods for model calibration.

4.1. Objective Functions

For the optimization, an objective function is needed as a quantitative measure of the error between the simulated and observed behaviors. Basically, the error measure can be any quantity that is not fixed in the simulation, such as the velocity, the acceleration, or the vehicle gap. In our implementation, we adopt the difference between simulated gap s_{sim} and real gap s_{data} as the error measure for the following reasons: when optimizing with respect to s, the average velocity errors are automatically reduced. However, when optimizing with respect to differences in velocity or acceleration, the errors in the distance may incrementally grow. Our intensive tests also show that the objective function with s is optimal. The integration of v and a to the function does not help improve the accuracy of the calibration.

In general, many error measures can serve as the objective function, such as absolute error and relative error. Since the absolute error measure systematically overestimates errors for large gaps at high velocities while the relative error measure is more sensitive to small distances s than to large distances, the mixed error measure is more robust. It can be seen as a combination of absolute error and relative error. As Kesting and Treiber did in [3], a mixed error measure is used in this paper:

$$F_{mix}[s_{sim}] = \sqrt{\frac{1}{\langle |s_{data}| \rangle} \left\langle \frac{(s_{sim} - s_{data})^2}{|s_{data}|} \right\rangle} \tag{4}$$

here $\langle . \rangle$ denotes the temporal average of a time series of duration ($1 \sim N$ frames in our problem). That is,

$$\langle s \rangle = \frac{1}{N} \sum_{i=1}^{N} s_i \tag{5}$$

s_{data} is the real gap between the subject vehicle and the vehicle in front, which can be obtained from the dataset. s_{sim} is the simulated gap, computed using the equation below:

$$s_{sim}(t) = x^l_{data}(t) - x_{sim}(t) \tag{6}$$

x^l_{data} means the leading car's real trajectory. The IDMM is used here to compute the following vehicle's simulated acceleration, which is then used to calculate its simulated trajectory $x_{sim}(t)$ by using Eqs. (11), (12). Each vehicle's initial state is assigned as $v_{sim}(t=0) = v_{data}(t=0)$ and $s_{sim}(t=0) = s_{data}(t=0)$, using the random values within the empirical range of parameters a, b, T, s_0, v_0, β.

4.2. Optimization with Adaptive Genetic Algorithm

Obviously, it is a nonlinear optimization problem to find a set of optimal IDMM parameters according to the given data set. Hence, conventional linear optimization methods cannot be directly applied here. Instead, a genetic algorithm (GA) is applied as a search heuristic to approximate the solution to the nonlinear optimization problem. SGA shows good performance in model calibration. However, its invariable parameters will conflict with the dynamic adaptation of GA. Usually, SGA converges fast to the sub-domain that contains the global optimum, and after that, it will probably become time consuming to locate the global optimum in local searching process. This is unsuitable for our situation as its computation would be heavy enough to require an offline fitting process. In this work, an adaptive GA is developed from simple GA and optimized for solving our multi-parameter optimization problem.

The improvements of AGA over SGA are as follows: (a). The constant crossover/mutation rate in the traditional GA is modified into an adaptive one in AGA, which may avoid the premature convergence of GA to a local optimum. (b). In order to accelerate the convergence speed, the elitist strategy is introduced in AGA after a round of selection, crossover and mutation operation.

The algorithm can be implemented as an iterative procedure that consists of a constant-size population of individuals. Each individual in the population represents a possible solution to the given problem. The genetic algorithm attempts to find the best solution to the problem by genetically breeding the population of individuals. The pseudo-code description of AGA is given in Algorithm 1.

Our adaptive genetic algorithm consists of the following steps:

1. Generation of initial population $P[0]$: This step is to set the initial points of searching and iteration. Suppose there are N individuals per population. Each individual represents a potential solution to the problem, i.e., a value set of $(v_0, T, a, b, s_0, \beta)$. They are all produced by adding random fluctuations to the empirical values. The individual should be encoded into the form that genetic algorithm can identify. One common approach is to encode solutions as binary strings: sequences of 1's and 0's, where the digit at each position represents the value of some aspect of the solution. Each binary string is called a chromosome in GA.

2. Fitness computing: As mentioned in Section 2.3, GA mimics the survival-of-the-fittest principle of nature to make a search process. The individuals with higher fitness value will have higher probability of being selected as candidates for further examination. Fitness represents the superiority or inferiority of an individual and Fitness function $F_{fitness}$ serves as the

Algorithm 1 Adaptive Genetic Algorithm

```
1:  gen ← 1
2:  Initialize the genetic parameters: Basicgen, $N$
3:  P[gen] ← GenerateInitialPopulation ($N$)
4:  while terminating conditions are not met do
5:      Evaluate the fitness of each individual in P[gen]
6:      S[gen] ← RWLSelection(P[gen])
7:      while $|S[gen]| \leq N$ do
8:          Select two individuals in S[gen]
9:          Compute the crossover rate $P_c$
10:         CM[gen] ← crossover (S[gen], $p_c$)
11:         Compute the mutation rate $P_m$
12:         CM[gen] ← Mutate (CM[gen], $p_m$)
13:         P[gen+1] ← ElitistSelect(P[gen], CM[gen])
14:     endwhile
15:     gen ← gen+1
16: endwhile
17: return the optimal parameter set
```

criterion of selection and elimination between generations. In our work, the optimization problem is stated in terms of minimization. In order to reflect the relative fitness of individual string, it is necessary to map the underlying natural objection function to fitness function. The adopted fitness mapping is presented as:

$$F_{fitness} = \frac{1}{1+F_{mix}} \tag{7}$$

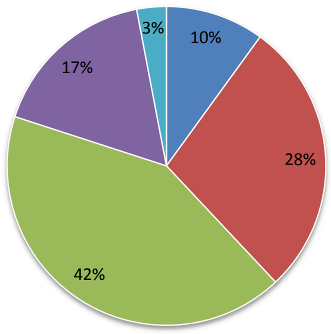
The equation ensures that fitness is non-negative and has a finite value for every error F_{mix}. The smaller the error F_{mix} is, the larger the fitness $F_{fitness}$ will be, which means chromosomes with larger fitness values possess larger probabilities to be selected.

3. Selection: Selection is the population improvement or "survival of the fittest" operator. Basically, it duplicates chromosomes with higher fitness and eliminates chromosomes with lower fitness. Roulette Wheel Selection Algorithm is a commonly used method to decide the quantity each individual duplicates itself to the next generation. The ith string in the population is selected with a probability proportional to its fitness $F^i_{fitness}$. Specifically, the probability for selecting the ith string is

$$P_i = \frac{F^i_{fitness}}{\sum_{i=1}^{N} F^i_{fitness}} \tag{8}$$

This could be imagined as a game of Roulette. Fig. 4 illustrates a roulette-wheel for each individual having different fitness values. The third individual has a higher probability of selection than any other. This roulette wheel selection scheme can be simulated easily. We then calculate the cumulative probability range CPR of each individual using Eq. (9).

$$CPR_i = \begin{cases} [0, P_i] & i=1 \\ CPR_{i-1} + (P_{i-1}, P_i] & \text{Otherwise.} \end{cases} \tag{9}$$



Individuals	P_i (%)
🟦 1	10
🟥 2	28
🟩 3	42
🟪 4	17
🟦 5	3

Figure 4: A roulette-wheel marked for five individuals according to their fitness values.

In order to choose n strings, n random numbers between zeros to one are created. The string whose cumulative probability range contains the random number is chosen to the matting pool. In this way, the individual with a higher fitness value will represent a larger range in the cumulative probability values and therefore has a higher probability of being copied into the matting pool.

4. Crossover and mutation: Crossover is a genetic operator that combines two chromosomes (parents) to produce a new chromosome (offspring), with the possibility that good solutions can generate better ones. Crossover occurs only with a user-definable probability p_c (called the crossover probability). Here we use Two-Point crossover method. The operator randomly selects two crossover points within a chromosome, and then interchanges the two parent chromosomes between these points to produce two new offspring. An illustration of the two-Point crossover operator is shown in Fig. 5.

After a crossover is performed, a mutation will take place. The mutation operator is used to maintain genetic diversity from one generation to the next. It alters one or more gene-bit values in a chromosome from its initial state. Mutation, similar to crossover, occurs according to a user-definable mutation probability p_m. Fig. 6 shows a schematic illustration of mutation. Mutation helps to prevent the population from stagnating at any local optima.

The power of genetic algorithm arises primarily from crossover and mutation. The choice of p_c and p_m critically affects the behavior and performance of GA. p_c controls the capacity of GA to converge near the global optimum after locating the region containing the optimum, and p_m controls the capacity of GA to explore new regions of the solution space in search of the global optimum. In the classic genetic algorithm, crossover and mutation work at a priori, constant probability. This can result in premature convergence of the GA to a local optimum. Therefore, it is essential to design an adaptive genetic algorithm that adapts itself to the appropriate crossover and mutation rates. Different crossover and mutation rate can traverse different search directions in the state space, thus affecting the performance of the applied genetic algorithm. In

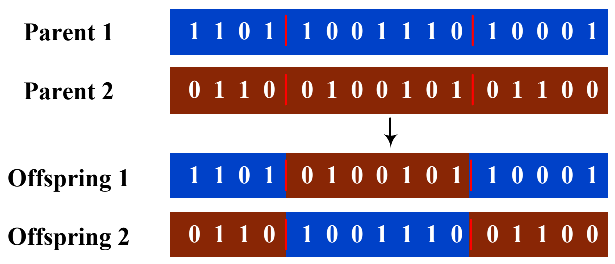


Figure 5: An illustration of Two-Point crossover.

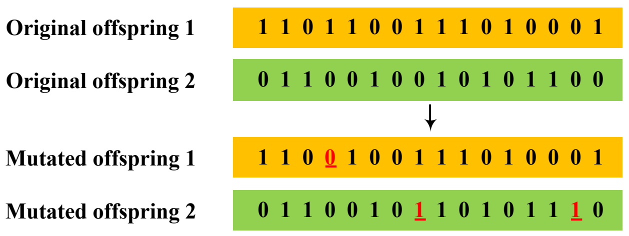


Figure 6: An illustration of mutation. The mutation operator simply inverts the value of the chosen gene-bits (0 goes to 1, and 1 goes to 0).

general, the rule should be: for high fitness solutions, lower values are assigned to p_c and p_m and higher values of p_c and p_m for low fitness solutions [25]. The expressions for p_c and p_m we adopted are given as:

$$p_c = \begin{cases} p_{c1} - \frac{(p_{c1}-p_{c2})(f'-f_{avg})}{f_{max}-f_{avg}} & f' > f_{avg} \\ p_{c1} & f' \leq f_{avg} \end{cases}$$
$$p_m = \begin{cases} p_{m1} - \frac{(p_{m1}-p_{m2})(f-f_{avg})}{f_{max}-f_{avg}} & f > f_{avg} \\ p_{m1} & f \leq f_{avg} \end{cases} \qquad (10)$$

where f_{max} is the maximum fitness value in the population, and f_{avg} is the average fitness value in the population. $f_{max} - f_{avg}$ is used to identify whether the GA is converging to an optimum or not. f' is the bigger fitness value in the two crossover individuals, and similarly, f is the fitness value of the individual to be mutated. $[p_{c2}, p_{c1}]$ and $[p_{m2}, p_{m1}]$ are respectively the valid ranges of crossover probability and mutation probability specified by users.

5. The elitist strategy: After a round of selection, crossover and mutation operation, a new generation will be built. However, because of the randomness of these genetic manipulations, it is likely that the best adaptive individuals in the current generation are destroyed later, which has a considerable impact on the operating efficiency and convergence of the genetic algorithm. So it is important to prevent promising individuals from being eliminated from the population between generations. To ensure the best individual is preserved, we introduce the elitist strategy. As shown in Fig. 7, we first retain the best half individuals (A, B) from parent generation, and then do selection, crossover and mutation to produce a new population. We choose the best individual E from the new population. If the fitness value of E is higher than A's, it suggests that the population has already been evolved toward the optimal solution. Otherwise, we will replace the worst half individuals (G, H) with the best half individuals in the parent generation (A, B). The implementation of the elitist strategy not only can guarantee that the optimal individuals will not be damaged by the genetic operators such as crossover and mutation, but also can guarantee the global convergence of genetic algorithm.

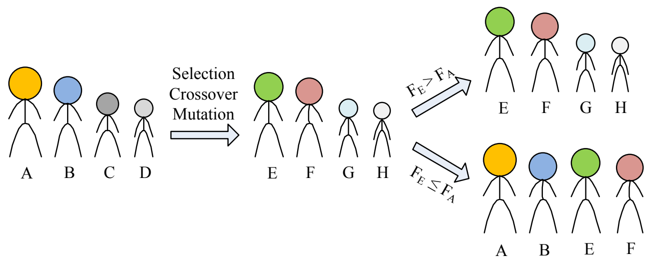


Figure 7: A process schematic diagram of the elitist strategy. The individual with bigger size represents owing the bigger fitness value.

6. Repeat 2, 3, 4, 5 until the predetermined terminal criterions are satisfied. The termination criterion is implemented as a two-step process. Initially, a fixed number of generations (called *Basicgen*) is iterated, which prevents the algorithm from returning a local optima. Then, the evolution terminated after convergence, which is specified by a fixed error for at least a given number of generations.

5. On-line Reconstruction and Simulation

Up to now, each vehicle's best-fit parameter set of IDMM has been found in the offline module by applying the described optimization method. The parameter set represents the driver's specific driving characteristics. Taking these learned variables, each car's initial state (position p and velocity v), and the trajectory data of the first vehicle in each lane as input, we can reproduce the traffic scene in the whole video (both the period used for training and the rest). In the simulation, each vehicle's acceleration can be calculated using Eqs. (1), (2), (3).

In each time step (ΔT), we update the vehicle's velocity according to kinematic principles:

$$v(t+1) = v(t) + a_{idmm}\Delta T \qquad (11)$$

and then update its position as following:

$$p(t+1) = p(t) + v(t+1)\Delta T \qquad (12)$$

In our implementation, we set $\Delta T = 1/30s$ (30 frames/second). We use the semi-implicit Euler integration to update the vehicle's velocity and position. All the simulations and experiments have shown that it can lead to stable results since the vehicle's acceleration value is independent of its position. The acceleration is calculated based on the vehicle's relative speed, acceleration and gap to its leading vehicle.

What's more, the driver's characteristics learned from the training data can be easily integrated into current large-scale traffic simulation systems based on microscopic models (in our experiment, the IDMM). There exist many research activities focused on planning an optimal trajectory strategy for each individual based on some optimization criteria, such as the minimum amount of (de-)acceleration, and the maximum distance to other cars to obtain safe and smooth motions, which are all too idealistic to reflect the real driving situations.

In this paper, we propose a new idea for traffic simulation, that is, sample-based traffic simulation. Instead of finding each vehicle's optimal trajectory, the goal of our approach is to reflect the most realistic traffic scene. According to each vehicle's trajectory data in the sample video, we get their real driving characteristics using the previously described approach. Taking these vehicles with personalized parameters as our example vehicles, we can build a vivid simulation scene in arbitrary size and duration based on the IDMM (see the simulation result in 6.2). In conclusion, traffic reconstruction is a scene reproduction of the sample video, and at a deeper level, our sample-based traffic simulation can be viewed as a scene extension in time and space.

6. Results

We have implemented and tested our method on a desktop PC equipped with Intel Core(TM)2 CPU 6320@1.86GHz, 4GB main memory (3.5GB available) and NVIDIA GetForce 8800 GTS graphics card.

6.1. Performance Analysis

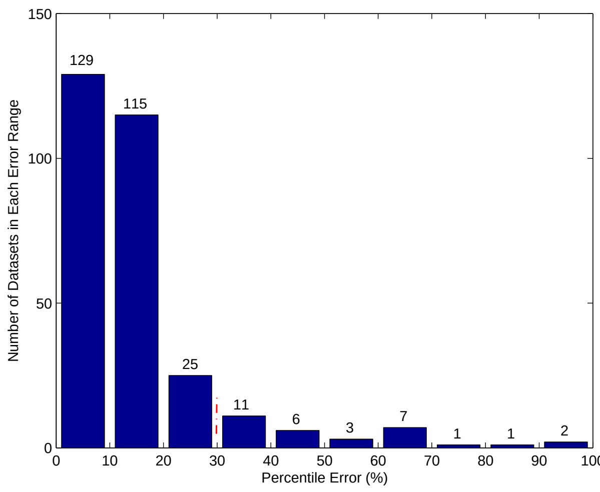


Figure 8: The learning error distribution of our algorithm (The total number of tested vehicles is 300).

The video we used to test is provided by NGSIM, captured from U.S. Highway 101 (Hollywood Freeway) in the Universal City neighborhood of Los Angeles, California, during 08:20AM-08:35AM. Each vehicle's trajectory data were recorded using NGSIM-VIDEO with the frequency of 10 frames per second. Since different roads in different areas may have different speed limits, it will result in different value ranges of a_0 and v_0 in our model. The local traffic rules and general driving habits are also influencing factors of the parameters' value range. A set of overall parameters' value range may lead to giant error when applied in some specific road sections. Thus, we choose to determine each parameter range for each road respectively. To U.S. Highway 101, the desired velocity v_0 is restricted to the interval $[15,40]$, the desired time gap T to $[1,5]$, the minimum distance s_0 to $[2,7]$, the maximum acceleration a to $[1.5,5]$, the comfortable deceleration b to $[0.1,3.5]$ and the adaptation factor β to $(0,3]$. We set the parameters of GAs with crossover probability range as [0.5,0.9], mutation probability range as [0.01,0.1].

Table 1: Personalized vehicle parameters of IDM and error rate.

	vehicle 88	vehicle 772
$Error\,[\%]$	1.83	5.91
$v_0\,[m/s]$	31.82	19.55
$T\,[s]$	1.28	1.19
$s_0\,[m]$	4.47	2.52
$a\,\left[m/s^2\right]$	0.54	0.61
$b\,\left[m/s^2\right]$	2.45	3.31
β	2.50	2.77

We randomly select 300 vehicles to test the performance of our algorithm and set the basic number of generations to 300, with 40 individuals per generation. Here we define the convergence as maintaining within a fixed error for at least 150 generations. The learning period is set as the first 300 frames of each car's trajectory data. By applying the optimization method described in Section 4, and measuring the difference between measured gap and simulated gap calculated from the calibrated parameters by the mixed error (Eq. (4)), we have found the best-fit parameters of IDMM to each vehicle's real trajectory data. Fig. 8 shows the resulting error distribution of our approach. Here the obtained error is defined in the range of $[0,30\%]$, which is consistent with typical error ranges obtained in the previous studies of model calibration ([3], [27]).

As is illustrated in Fig. 8, among the 300 tested vehicles, 269 of them result in the error rate less than 30%, whereas only 31 tested vehicles lead to the error larger than 30%. This shows that the simulated behaviors of most vehicles are approximate to the real trajectory at an acceptable error rate, which implies that our algorithm is a convincing method to learn each vehicle's specific driving habits and further produce realistic traffic simulation scenes. We note that the obtained data from NGSIM-VIDEO has noise because of some objective and subjective influence factors in the data-acquisition procedure. This is a factor which may lead to notable errors in our learning results. Another factor for the fit error may come from the non-constant driving style of drivers.

For detailed illustration, we show the learning results of Vehicle 88 and Vehicle 772 in Table 1, and compare our reconstruction result with the real data of these two cars respectively in Fig. 9. For both the two datasets, we take $0 \sim 300th$ frames of the trajectory data as the learning sample. The GA heuristic has found the best match between the recorded measures

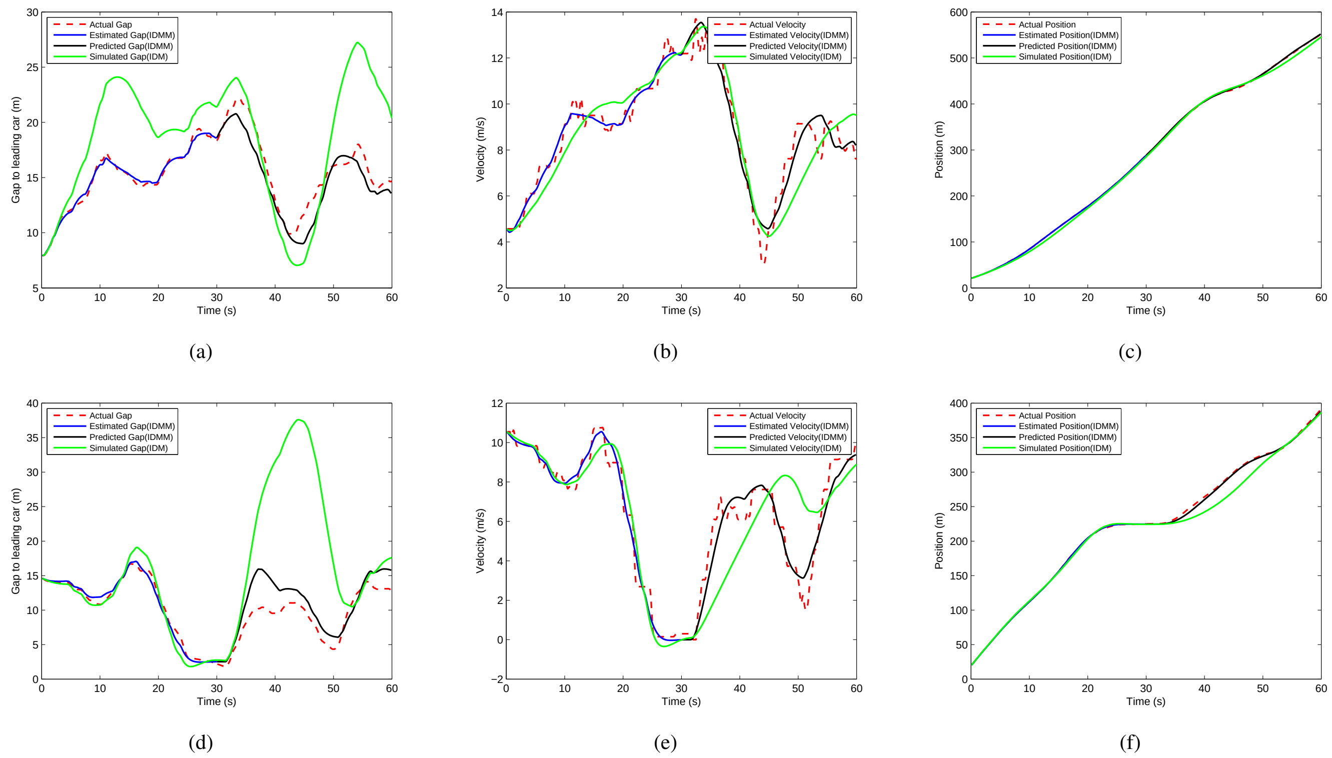


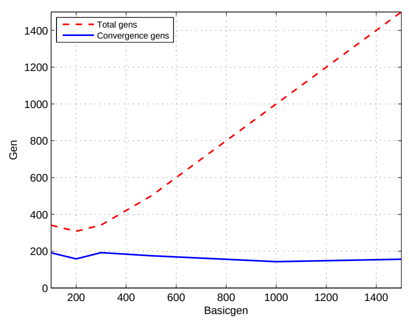
Figure 9: Comparison of simulated and observed trajectories of vehicle 88 (the top three images) and vehicle 772 (the bottom three images). The observed trajectories are all extracted from the video sample. The red line represents the real data. For comparison, the blue and black lines respectively represent the training and the predicting period with the IDMM. The greens show the simulated results using IDM.

and the simulated ones for the parameter values presented in Table 1. The corresponding mixed errors are 1.83% for Vehicle 88 and 5.91% for Vehicle 772 separately. As shown in Table 1, the resulting model parameters vary from one vehicle to another because of the different driving situations and driving habits. The comfortable deceleration of Vehicle 772 is 3.31 m/s^2, larger than Vehicle 88's 2.45 m/s^2 deceleration value, which means Driver 772 is likely to brake sharply, while Driver 88 is much more careful. On the other hand, this also leads to Vehicle 772's shorter reaction time than Vehicle 88, which is just in accordance with our learning result for T.

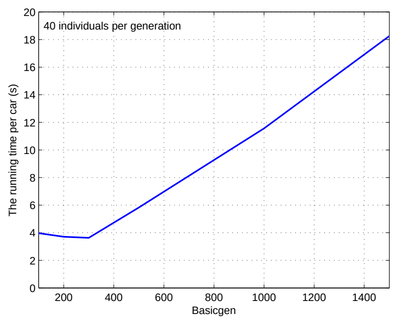
Fig. 9 compares these two vehicles' dynamics resulting from the computed parameters in Table 1 with their empirically measured values. It plotted all the 0∼600*th* frames measured and simulated data. The red lines represent the measured data, covering the entire timeline, and the blue and black lines both stand for the simulated data using IDMM. Selecting each vehicle's 0*th* to 300*th* frame data as learning sample, the assessments of learning results are shown in the blue parts. Fig. 9(a) and (d) show the comparison on the gap to the leading vehicle. It can be seen that the blue part is basically close to the red part, which proves that our proper chosen objective function works well. The green lines show the learning results using IDM. It further indicates IDMM is acceptable to approximate drivers' real driving behaviors, and our GA-based optimization approach is suitable to find the best match between the measured trajectory and the microscopic traffic model.

We use the resulting credible personalized parameters to simulate vehicles' behaviors in the subsequent frames, and the relevant data of 300*th* to 600*th* frames is plotted in the black line part of Fig. 9. we can see 301 to 600 simulation results are still basically in accordance with the real data, indicating that our approach can be highly predictive for each vehicle's driving behavior. Since the error measure is chosen as the vehicle gap when constructing the objective function, the accumulated error of our algorithm is suppressed to some extent. Fig. 9(b) and (e) respectively show Vehicle 88 and Vehicle 772's comparison results of momentary velocity. We can see from these two figures that the velocity error has been automatically reduced while optimizing with respect to distance. Moreover, in order to better reflect the performance of our approach, Fig. 9(c) and (f) intuitively plot the two cars' position comparison results. Overall, the tests in Fig. 9 have thoroughly validated that our method is accurate enough to be applied in traffic reconstruction and sample-based traffic simulation.

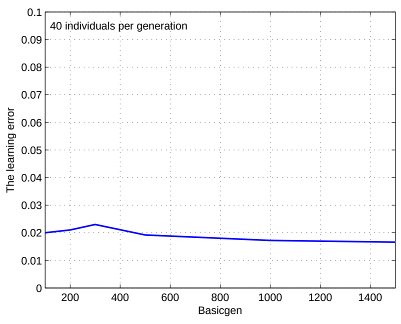
Adaptation to the environment: As shown in Fig. 9(a) and (d), the blue and black lines show the learning results using IDMM, while the green lines show the learning results with IDM. It is not hard to see that, without considering the adaptation factor to the surroundings, IDM has lower performance than IDMM. The introduction of β successfully reduces the learning error and makes the driving model better adapt to the real-world data. IDM can reflect the vehicle's basic driving habits, but is unable to show the respond to the changes in the



(a)

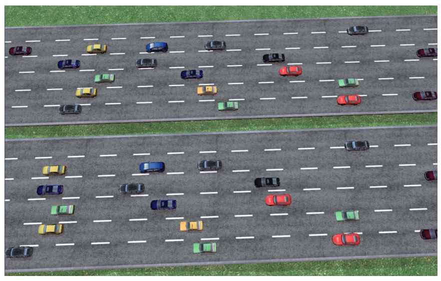


(b)

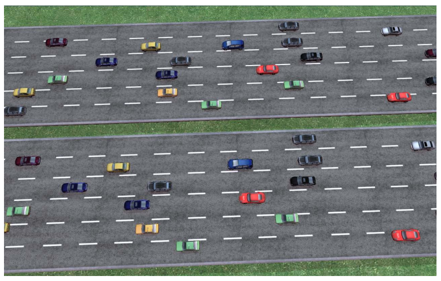


(c)

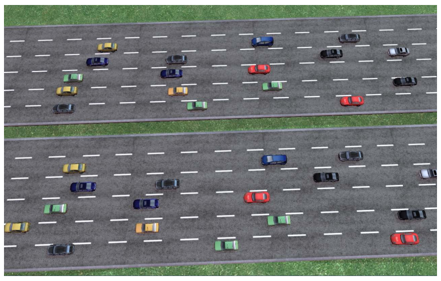
Figure 10: Impact of Basicgen on the performance of AGA.



(a) 500th frames



(b) 1500th frames



(c) 3000th frames

Figure 11: The representative frames of the original data and the reconstruction data. (a) 500th frames (b) 1500th frames (c) 3000th frames. In all pictures, the top road shows the real traffic flow while the below one shows our reconstruction result.

environment in a timely manner. This comparison, on the other hand, shows the adaptation factor β is not a negligible factor to keep in accordance with the real driving behavior.

Convergence of AGA: For the 300 vehicles we tested, Table 2 presents the average performance of the our adaptive genetic algorithm, and compares it with the simple genetic algorithm. It can be seen that, using our AGA, all vehicles are all perfectly converged within 300 generations on average, and the total iterative generations are below 400 generations. In contrast, the simple genetic algorithm has a poor performance for our specified convergence rules. The search can last more than 1000 generations and the convergence percent is only 14%. The employment of static p_c and p_m is part of the reason that SGA fails to promise convergence speed or even convergence in some cases. In addition, the randomness of crossover and mutation also impacts the operating efficiency and convergence of the genetic algorithm. On the contrary, in our adaptive genetic algorithm, the adaptive crossover and mutation rate and elitist strategy greatly accelerate the convergence speed and improve the overall search capabilities. It is unlikely to result in a local optimal solution, and the convergence speed is much faster than that in SGA. All the tests show that AGA can achieve a much higher accuracy than SGA.

Impact of Basicgen: Fig. 10 shows the Basicgen's impact on the performance of AGA. In Fig. 10(a), we can see that, with the increasing of Basicgen, the convergence generations maintain in the same level. The total running generation numbers maintain in the range of [300, 400] when Basicgen is set below 300 whereas it grows linearly when Basicgen is set bigger than 300. The similar rule of the running time can be found in Fig. 10(b). Fig. 10(c) proves that, for our terminal criteria, simply increasing Basicgen has no effect on the learning error. It also gives grounds for our setting of the Basicgen as 300 in the above tests.

Table 2: Performance comparison between our AGA and Simple GA.

	Our AGA	Simple GA
converge gens per car	217	1000+
total gens per car	388	1000+
convergence percent	100%	14%
calculate time per car	3.76s	34.01s

Timing performance: Because the personalized driving behavior learning is computed offline, our method adds no overhead to the overall simulation runtime. Our offline training time is equal to the convergence time of AGA. Table 2 shows the timing performance of our offline training. For 300 tested vehicles, the average training time is 3.76 seconds. Such a time is acceptable for offline processing and much shorter than that required with SGA (34.01 seconds shown in Table 2). In Fig. 10(b), we plot the average training time as a function of the iteration Basicgen. Note that our aim is to learn personalized driving characteristics from an input video instead of reconstructing traffic flows in real time from the input video.

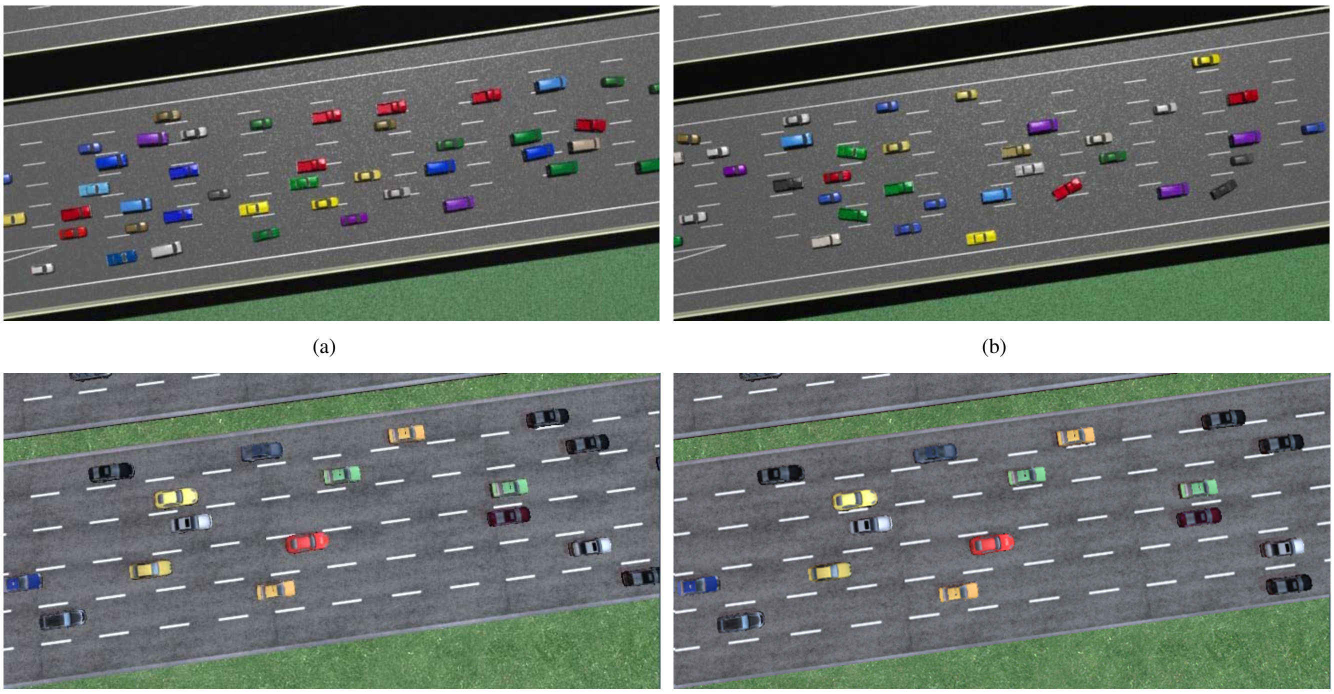


Figure 12: a) Original I-80 vehicle trajectory data. b) Reconstructed I-80 vehicles using Sewall's approach [1]. c) Original US-101 vehicle trajectory. d) Reconstructed US-101 vehicles using our approach.

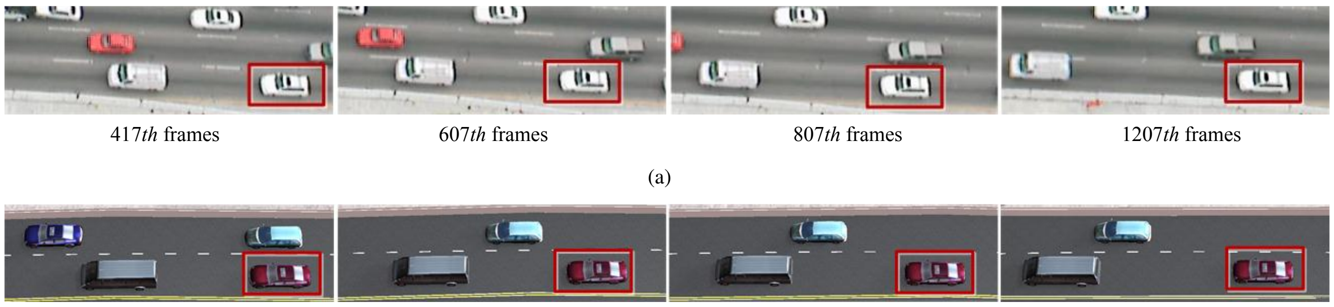


Figure 13: Snapshots of a single car's (in red rectangle) behavior in the sample video (a) and our simulation scenario (b). The car's 10∼800*th* frame behaviors in our simulation are similar with the marked car's 417∼1207*th* frame behaviors in the video.

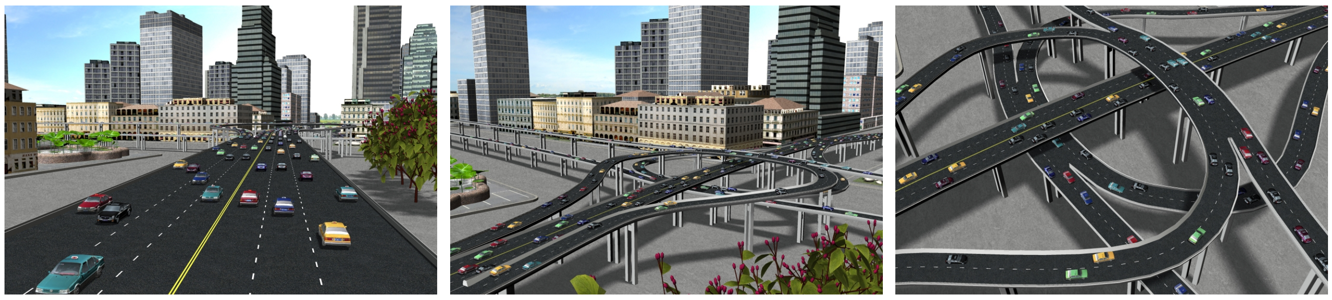


Figure 14: Some traffic simulation scenarios generated using our sample-based method.

6.2. *Simulation Results*

We have built a typical highway road network and visualized the various traffic scenarios using our approach. Our system can simulate each vehicle intelligently as if a real driver was in it.

Representative frames of the reconstructed traffic flows are shown in Fig. 11. It can be seen that our reconstruction is a high approximation to the real traffic flows, and some of the vehicles even have the same behaviors as the real ones. On the other hand, it proves that the cumulative error of our approach is always maintained in an acceptable scope over time. To the best of our knowledge, in computer graphics, Sewall's virtualized traffic [1] is currently the only work on traffic reconstruction, we compare our visualized result to Sewall's in Fig. 12. As the main goal of our system is to reflect each driver's characteristic driving habits, which is totally different from Sewall's reconstructing traffic flows from discrete data obtained by sensors placed alongside a road, we only make a visual comparison on the reconstruction accuracy of real-world traffic flows. Fig. 12(a) and (b) are Sewall's results which are provided in their article [1]. Fig. 12(c) and (d) show our results. Our benefit mainly lies in three points:

1) Our approach can achieve a high degree of accuracy in a smaller range of reconstruction while Sewall's approach is only accurate in a coarse level. Their reconstruction has the same vehicle position with the original data at every sensor point (200 to 400 meters apart).

2) Our video-based method has wider applications than Sewall's sensor-based approach. Our learned driving characteristics can be applied to other scenarios, regardless of time and space. In this sense, our method is more flexible in applications.

3) What's more, our video-based method is more cost-effective and convenient to use than Sewall's sensor-based method. Although traffic sensors can access data from monitoring sites directly, they are unsuitable for an ordinary user due to the high expense and hard maneuverability on a long road. In contrast, our video-based method provides a much cheaper solution with less but acceptable accuracy.

In addition, Fig. 14 shows some traffic simulation scenarios generated using our sample-based method. For the selection of model variables, our sample-based method provides a theoretical basis and has more practical guiding significance, comparing with the previous simulation method. By applying the vehicle's specific driving characteristics learned from the sample video in other traffic environments, our approach can simulate a realistic traffic scenario with various driving behaviors that look like those in the sample video. In contrast, in the previous method, the vehicle's driving behaviors are determined by randomly generated parameters. In order to show variety, they just enlarge the range of random values without any realistic reference. Also, in order to achieve realistic results around the stop-and-go road regions, they have to spend many efforts on regulating individual vehicle's parameters manually. This further indicates that our approach has a great realistic significance and reference value on assessment and improvement of traffic networks. Fig. 13 shows some comparison snapshots of the behavior for an individual driver in the sample video and our simulation scenario. Personalized driving characteristics are obvious in Fig. 13 although they cannot be easily found in the accompanying demo.

7. Conclusions and Future Work

This paper presents a new approach for traffic reconstruction and sample-based simulation, in which each vehicle's specific driving characteristics are learned from the trajectory data exacted from the video sample provided by users. We introduce an adaptive genetic algorithm to search for each vehicle's optimal parameter set of IDMM. Our adaptive genetic algorithm outperforms existing methods for model calibration. The adaptive crossover and mutation rate and elitist strategy greatly accelerate the convergence speed and improve the overall search capabilities. What's more, because of the use of IDMM, our system can describe the adaption of drivers to the surrounding traffic situation.

To the best of our knowledge, in the computer graphics community, this is the first attempt at presenting each vehicle's real specific driving characteristics and simulating a virtual traffic flow that behaves similarly to the real traffic in the input video.

Besides the AGA, other nonlinear optimization methods, such as PSO, are suitable candidates for the offline learning as well. Our offline training process can be further accelerated because each driver's driving characteristics can be calculated in parallel with the input video available. In our current implementation, the vehicle trajectory data is obtained using the software NGSIM-VIDEO. The defects of the software itself and the shortcomings in manual operations both may bring about the noise in data. We are now developing our own system on vehicle detection and tracking. In addition, we mainly focus on the de-acceleration strategy learning. Little work has been done to lane-changing behaviors because of its complex nature. In our future work, an extension would be to learn each vehicle's lane changing habits to overcome the limitation, which will lead to a more vivid traffic reconstruction and simulation.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant no. 61272298), Zhejiang Provincial Natural Science Foundation of China (Grant no. Z1110154), the National Key Basic Research Foundation of China (Grant no. 2009CB320801), the China 863 program (Grant no. 2012AA011503), and the Major Science and Technology Innovation Team (Grant no. 2010R50040).

References

[1] J. Sewall, J. van den Berg, M. C. Lin, D. Manocha, Virtualized traffic: Reconstructing traffic flows from discrete spatio-temporal data, IEEE Transactions on Visualization and Computer Graphics 17 (1) (2010) 26–37.

[2] J. Sewall, D. Wilkie, M. C. Lin, Interactive hybrid simulation of large-scale traffic, ACM Transactions on Graphics (TOG) 30 (6) (2011) 135.

[3] A. Kesting, M. Treiber, Calibrating car-following models by using trajectory data: Methodological study, Transportation Research Record: Journal of the Transportation Research Board 2088 (4) (2008) 148–156.

[4] M. Treiber, D. Helbing, Memory effects in microscopic traffic models and wide scattering in flow-density data, Physical Review E 68 (4) (2003) 046119.

[5] M. J. Lighthill, G. B. Whitham, On kinematic waves. ii. a theory of traffic flow on long crowded roads, in: Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 1955, pp. 317–345.

[6] H. J. Payne, Models of freeway traffic and control, Mathematical Models of Public Systems 1 (1) (1971) 51–61.

[7] G. B. Whitham, Linear and nonlinear waves, Wiley, New York, 1974.

[8] A. Aw, M. Rascle, Resurrection of second order models of traffic flow, SIAM Journal of Applied Math 60 (3) (2000) 916–938.

[9] H. M. Zhang, A non-equilibrium traffic model devoid of gas-like behavior, Transportation Research Part B 36 (3) (2002) 275–290.

[10] J. Sewall, D. Wilkie, P. Merrell, M. C. Lin, Continuum traffic simulation, Computer Graphics Forum 29 (2) (2010) 439–448.

[11] D. L. Gerlough, Simulation of freeway traffic on a general-purpose discrete variable computer, PhD thesis, UCLA (1955).

[12] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, Y. Sugiyama, Dynamic model of traffic congestion and numerical simulation, Physical Review E 51 (2) (1995) 1035–1042.

[13] M. Treiber, D. Helbing, Microsimulations of freeway traffic including control measures, Automatisierungstechnik 49 (2001) 478–484.

[14] J. Shen, X. Jin, Detailed traffic animation for urban road networks, Graphical Models 74 (5) (2012) 265–282.

[15] K. H. Lee, M. G. Choi, Q. Hong, J. Lee, Group behavior from video: a data-driven approach to crowd simulation, in: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association, 2007, pp. 109–118.

[16] E. Ju, M. G. Choi, M. Park, J. Lee, K. H. Lee, S. Takahashi, Morphable crowds, ACM Transactions on Graphics (TOG) 29 (6) (2010) 140.

[17] S. J. Guy, S. Kim, M. C. Lin, D. Manocha, Simulating heterogeneous crowd behaviors using personality trait theory, in: Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, ACM, 2011, pp. 43–55.

[18] L. Chu, H. X. Liu, J.-S. Oh, W. Recker, A calibration procedure for microscopic traffic simulation, in: Intelligent Transportation Systems, Vol. 2, IEEE, 2003, pp. 1574–1579.

[19] J. Hourdakis, P. G. Michalopoulos, J. Kottommannil, Practical Procedure for Calibrating Microscopic Traffic Simulation Models, Transportation Research Record 1852 (2003) 130–139.

[20] R. L. Cheu, X. Jin, K. Ng, Y. Ng, D. Srinivasan, Calibration of fresim for singapore expressway using genetic algorithm, Journal of Transportation Engineering 124 (6) (1998) 526–535.

[21] A. Kesting, M. Treiber, Calibration of car-following models using floating car data, in: Traffic and Granular Flow 07, Springer Berlin Heidelberg, 2009, pp. 117–127.

[22] H. Ceylan, M. G. Bell, Traffic signal timing optimisation based on genetic algorithm approach, including drivers routing, Transportation Research Part B: Methodological 38 (4) (2004) 329–342.

[23] S. B. L. Sadek, A. W., M. J. Demetsky, Dynamic traffic assignment: Genetic algorithms approach, Transportation Research Record: Journal of the Transportation Research Board 1588 (1) (2007) 95–103.

[24] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, 1st Edition, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[25] M. Srinivas, L. Patnaik, Adaptive probabilities of crossover and mutation in genetic algorithms, Systems, Man and Cybernetics, IEEE Transactions on 24 (4) (1994) 656–667.

[26] NGSIM, Next generation simulation program, http://ngsim-community.org/ (2008).

[27] P. Ranjitkar, T. Nakatsuji, M. Asano, Performance evaluation of microscopic traffic flow models with test track data, Transportation Research Record: Journal of the Transportation Research Board 1876 (1) (2004) 90–100.