Team Control Number

For office use only

T1 ________________

T2 ________________

T3 ________________

T4 ________________

29911

Problem Chosen

A

For office use only

F1 ________________

F2 ________________

F3 ________________

F4 ________________

Summary

The keep-right-except-to-pass (KRETP) rule has been adopted by many countries around the world, but does this rule actually make our transportation system more efficient? This report aims to analyze this rule along with several other traffic regulations.

Using a discrete cellular-automaton (CA) model and a continuum model, we can simulate real-life traffic situation on freeways via the *Monte Carlo* method and PDE system respectively. Through comparison with other two traffic rules, we obtain the conclusion that the KRETP rule is rather effective.

First we define three parameters---traffic flow, safety index and average energy consumption (AEC) to evaluate the performance of the KRETP rule under various vehicle density. By calculating the optimal maximum velocities during light and heavy traffic, we obtain the influence of under-posted and over-posted speed limits. We also assert that our model can be transferred in "left-most" countries with a simple change of orientation.

Then we introduce two other traffic rules----the "Slow-Cars-To-Right" (SCTR) rule and the "Free Driving & Free Overtaking" (FDFO) rule. By comparing these three rules in terms of our pre-defined parameters, we confirm KRETP rule's superiority and provide strategic advice for future freeway construction.

Next, under the control of an intelligent system, a "median" optimization method is proposed to improve the overall quality of freeway transportation system. According to simulation result, our optimization method does improve the performance in terms of all three parameters.

Finally, we discuss upon several defects of our model that require further research.

Keywords: KRETP Rule, CA Model, Continuum Model, *Monte Carlo* Method, Traffic Flow, Safety Index, Average Energy Consumption (AEC), Optimization

The Keep-Right-Except-To-Pass Rule

Contents

1 Introduction

Traffic rule plays an essential role in a nation's transportation system. An optimal traffic rule can dramatically enhance the capacity and efficiency of the transportation network, providing common citizens with tremendous convenience. In this report, we will mainly focus on one such rule which requires automobiles to stay in the right lane unless they have to overtake.

America first enacted the "keep-right-except-to-pass" law in New York State in 1804, followed by many other countries in the following decades. [1] Up to now, most countries are "right-handed" countries with very few exceptions such as the UK and Australia. On American multilane freeways, for example, drivers are required to drive in the right-most lane unless they need to overtake, in which case they move one lane to the left and then switch back to the right-most lane. See in **Figure 1**.

Despite the long history of this rule, there has been little scientific inquiry into the effect of this rule. In such circumstances, a systematic analysis of this rule's realistic effect, including the change in traffic flow, safety index and energy consumption.

More specifically, we are expected to examine the following issues in our report:

- Analyze the effect of the "keep-right-except-to-pass" rule in various terms under different vehicle densities (light and heavy traffic);
- Calculate the optimized speed to maximize a weighted function of the traffic flow, safety index and average energy consumption.
- Discuss the role of under-posted and over-posted speed limits and the transferability of our model in "left-handed" countries where driving on the left lane is the norm;
- Propose other possible freeway traffics rules and compare them with the "keep-right-except-to-pass" rule;
- Discuss the transferability of our model in "left-handed" countries where driving on the left lane is the norm;
- Design optimization method to enlarge traffic flow when equipped with an intelligent system.

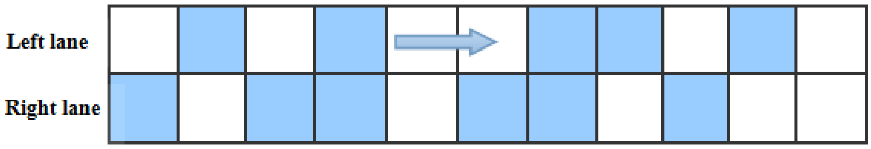

Figure 1. Illustration of "Keep-Right-Except-To-Pass" Rule

2 Assumptions

- Assuming automobile to move in straight lines;
- Assuming that jams only occur because of gradual piling of automobiles;
- Without consideration of sudden crashes that will influence the behavior of subsequent vehicles;
- Assuming that the parallel movement of a vehicle into another lane takes no time;
- Assuming that all the drivers on the freeway are abide by the traffic rules;
- Assuming that the vehicles have the same length and mass;
- Assuming that freeways have no extra entrance or exit.

3 Modeling for Right-Most Rule

In order to analyze the effect of the "right-most" rule, first we have to establish a model to simulate the traffic flow on the multilane freeway. Then we add certain driving and lane-changing rules to make sure that in our simulation the automobiles are abide by the "right-most" rule.

3.1 Discrete Modeling for Right-Most Rule

Discrete models treat both time and the position of vehicles as discrete quantities and simulate the moving of automobiles step by step. We start out simulation with a single kind of vehicle (single maximum speed) on a double-lane freeway (right lane is slow lane and left lane is fast lane) and then extend to two kinds of vehicles, namely, fast car and slow car (two maximum speeds), which is closer to realistic situation.

3.1.1 Model Establishment

- **Vehicles of a Single Kind**

Among the various discrete models, we adopt the "Particle-Hopping" model, which was initially formulated by Nagel and Schreckenberg (NS) to idealize the movement of vehicles as the discrete "hopping" of particles. **Figure 2** is a vivid illustration of our model. We treat the double-lane freeway as a two-column lattice and each vehicle contains exactly one lattice.

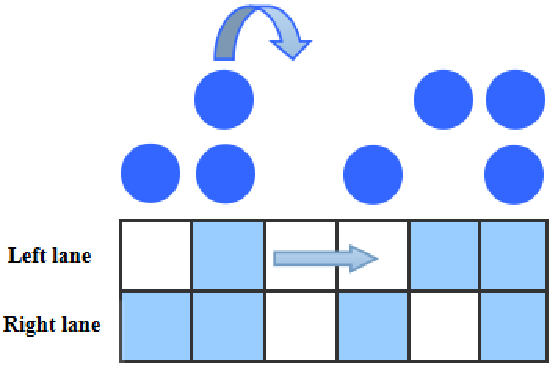

Figure 2. Representation of the "Particle-Hopping" Model

We apply the *Monte Carlo* method to simulate the traffic flow on the double-lane freeway.[2] The quantities involved in the simulation are shown in **Table 1.**

Notation	Definition
$X(k)$	The position of car numbered k
$V(k)$	The velocity of car numbered k
$\Delta X_p^f(k)$	The gap in front of car numbered k in the present lane
$\Delta X_p^b(k)$	The gap behind car numbered k in the present lane
$\Delta X_o^f(k)$	The gap in front of car numbered k in the other lane
$\Delta X_o^b(k)$	The gap behind car numbered k in the other lane
$V_{max}(k)$	The maximum velocity the k^{th} car can achieve on the freeway
Q	The traffic flow(number of cars passed per lane per unit time)
α_{safe}	The safety index
L	The length of the lane during simulation*
T	The duration of the simulation*

P_d	The probability of a car decelerating*
P_c	The probability of a car changing lane*
n	The density of vehicles *
N	The number of car during simulation $N = 2nL$

Table 1. Definition of Notation in "Particle-Hopping" Model

(Terms marked with an asterisk (*) require pre-evaluation before the simulation begins)

Before the simulation begins, we need to evaluate the following five parameters: number of cars N, the length of lane L, the duration of the simulation T, the maximum velocity vehicles can achieve V_{max}, the probability for a driver to decelerate during each unit time P_d and the probability for a driver to change lane P_c if the current situation qualifies the rule of changing lanes.

After these parameters have been evaluated, we set the position of each car $X(n)$ randomly in the double-lane freeway with half cars on the left (fast) lane and the other half on the right (slow) lane and the velocity of each car $V(n)$ randomly between $[V_{max}/2, V_{max}]$. We set the entire freeway to be a loophole so that once a car reaches the last lattices it will re-enter the double-lane freeway in the first lattices. Under such conditions, the density of the vehicles remain unchanged throughout the simulation.

After the simulation begins, at each time unit the driver will either remain in the same lane or change lane if situation permits. The rule of acceleration and deceleration is as follows:

$(i)\ \ If\ V(n) < V_{max}, then\ V(n) = V(n) + 1;$
$(ii)\ If\ ran\ f < p_d, then\ V(n) = V(n) - 1;$

Table 2. The Rule of Acceleration and Deceleration

where $ran\ f$ is a random number between 0 and 1 generated at each time unit.

The permission rule for changing lane is much more complex and comprised of several basic rules. It is derived from *Wagner*'s according to the statistics gathered from a Germany freeway where "right-most" rule is adopted:[3]

Rule safety	**Rule stay except blocked (Rule #0)**	**Rule change when possible (Rule #1)**
(i)$\Delta X_p^f(k) > 0$, enough space in front	(i) $\Delta X_p^f(n) < V_{max} + 1$, not enough space in front	(i)$\Delta X_o^f(n) > \Delta X_p^f(n)$(more space on the other lane) or $\Delta X_o^f(n) > V(k)$(enough space on the other lane)
(ii) The nearest neighbor site in the other lane is empty.	(ii) $\Delta X_o^f(n) > \Delta X_p^f(n)$, more space on the other lane	(ii) *ran* f < *Pc*
(iii) $\Delta X_o^b(n) > V_{max}$, enough space to the next car on the other lane	(iii) *ran* f < *Pc*	

Table 3. The Basic Rules

$right \rightarrow left$		$left \rightarrow right$	
(i)	Rule safety	(i)	Rule safety
(ii)	Rule #0	(ii)	Rule #1

Table 4. The Rule of Lane Changing corresponding to the “rightmost” rule

Our *Monte Carlo* simulation has three outputs: traffic flow Q and safety index α_{safe} and the energy cost E_0. For traffic flow Q, we choose a fixed point (in our simulation we choose the end point of the lattices since the freeway is a loop) to count the total number of cars passing that point denoted as N_{total} within the duration of the simulation T, then we use the following equation to calculate Q:

$$Q = \frac{N_{total}}{T}.$$

As for α_{safe}, we make the assumption that safety index is proportionate to the reaction time of all the drivers on the freeway throughout the entire simulation. We take the proportionate coefficient to be 1 for convenience:

$$\alpha_{safe} = \frac{\sum_T \sum_1^N [\exp(-\Delta X_p^f(n)/V(n))]}{N \cdot T}.$$

In **Table 5**, we present the specific steps of our *Monte Carlo* simulation:

Input	Length of the lane, ***L***. Duration of the simulation ***T***.

	Vehicle density $\boldsymbol{n}$. Deceleration probability $\boldsymbol{P_d}$. Lane changing probability $\boldsymbol{P_c}$.
Output	Traffic flow Q. Safety index α_{safe}. Energy cost E_0
Step 1	Randomly generate the initial position, speed and max speed of vehicle i $X(k)$, $V(k)$ and $V_{max}(k)$.
Step 2	Repeat Step 3 ~ 16 T times.
Step 3	Repeat Step 4~5 for each vehicle
Step 4	Apply the safety rule to vehicle i, skip Step 5 if vehicle i doesn't pass the rule
Step 5	Apply either change lane rule #0 or change lane rule #1 to vehicle i based on which lane it is on, its current speed and the current model, then decide whether it should change its lane.
Step 6	Update each vehicle to its new lane.
Step 7	Repeat Step 8~16 for each vehicle number k.
Step 8	Let the expected new speed $V'(k) = V(k) + 1$
Step 9	If $\Delta X_p^f(k) < V'(k)$, let $V'(k) = \Delta X_p^f(k) - 1$.
Step 10	If $V'(k) > 0$, generate a random number in [0, 1] and check whether it is smaller than P_d, let $V'(k) = V'(k) - 1$ if so.
Step 11	Let $X'(k) = X(k) + V'(k)$
Step 13	$$\alpha_{safe} = \alpha_{safe} + \exp(-\frac{\Delta X_p^{f}{}'(k)}{V'(k)})$$
Step 14	If $X'(k) \geq L$, then $X'(k) -= L$, $Q = Q + 1$
Step 15	If $V(k) < V'(k)$, $E_0 += V(k) + V'(k)$
Step 16	Let $X(k) = X'(k)$, $V(k) = V'(k)$
Step 17	$E_0 = \frac{E_0}{Q}$, $\alpha_{safe} = \frac{\alpha_{safe}}{Q}$, $Q = \frac{Q}{T}$
Step 18	Output and halt.

Table 5. *Monte Carlo* Procedures

- **Vehicles of Two Kinds (Fast Cars & Slow Cars)**

Next, we consider a more complex situation in which cars are classified into fast cars and slow cars with respective maximum speed V_{max}^f and V_{max}^s, with other conditions and rules unchanged.

Next, we apply a similar *Monte Carlo* algorithm to simulate the traffic flow with two kinds of vehicles and output corresponding flow Q and safe index α_{max}.

3.1.2 Parameter Evaluation

- The length of the lattices $L = 2048$;
- The vehicle density $n = 0.02, 0.04, 0.06, 0.08, 0.1, 0.13, 0.16, 0.2, 0.25, 0.3$;
- The number of the total vehicles $N = 2nL$;
- The duration of the simulation $T = 4096$;
- The deceleration probability $P_d = 0.1$;
- The change lane probability $P_c = 0.7$;
- In the case of vehicles of single kind, set maximum velocity $V_{max} = 5$;
- In the case of vehicles of two kinds, set maximum velocity of fast car $V_{max}^{f} = 5$, set maximum velocity for slow car $V_{max}^{s} = 3$;
- The ratio of fast car to slow car is 4:1, the initial number of vehicle in each lane is 1:1.

3.1.3 Model Solution and Analysis

We mainly analyze the influence of the right-most rule on three parameters: Traffic flow Q, safety index α_{safe} and average energy consumption E_0. Then we assign a relative weight to each parameter to calculate an optimal velocity during light traffic and heavy traffic. We will only present the simulation result of the two-kind vehicle model.

- **Traffic Flow**

As we have discussed before, we define the traffic flow Q as the number of cars passing a fixed point per unit time, or

$$Q = \frac{N_{total}}{T}$$

In **Figure 3**, we present the value of traffic flow Q under different vehicle density n.

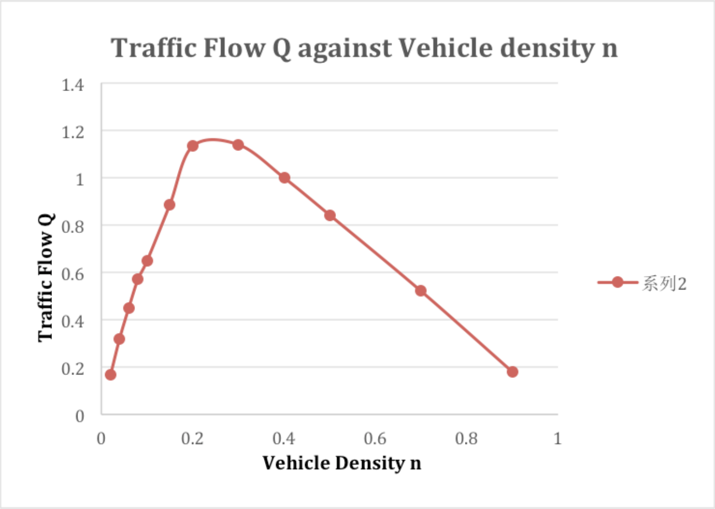

Figure 3. Relationship of Traffic Flow Q and Vehicle Density n

Apparently, the pattern in **Figure 3** accords with realistic situation qualitatively. When the vehicle density on a freeway is relatively low, slight increase of vehicle, or in other words, vehicle density will cause traffic flow to rise dramatically because according to our rules vehicles accelerate very easily. In our simulation, the peak value of flow Q occurs when density is around 0.2. After the climax, the traffic flow will tend to drop as vehicle density continues to rise because more likely car jams will obstruct vehicles from accelerating freely. Finally, when the density approaches 1, in other words, the freeway is almost "full", vehicles could barely move thus the flow approaches 0.

- **Safety Index**

Notice that the longer reaction time is, the safer the situation will be. Thus according to our definition of the safety index

$$\alpha_{safe} = \frac{\sum_T \sum_1^N [\exp(-\Delta_p^f X(n)/V(n))]}{N \cdot T}$$

a larger value of α_{safe} indicates a more dangerous situation. We again plot safety index α_{safe} against vehicle density n. The result is shown in **Figure 4**.

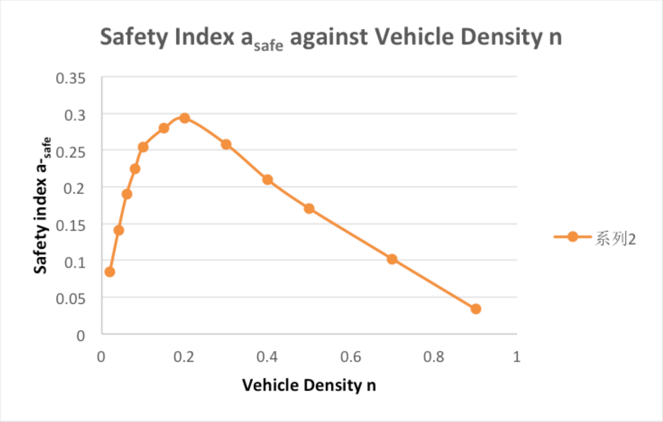

Figure 4. Relationship of Safety Index α_{safe} and Vehicle Density n (lower is safer)

It is easy to understand the pattern in **Figure 4**. When the vehicle density is relatively small, accidental risk rises as the number of car increases because vehicles are likely to attain high velocities. After the peak value, the risk drops as density rises because though the number of vehicle rises, the average velocity decreases dramatically. Since most vehicles drive much slower, the overall situation becomes actually safer. Finally, when all the vehicles cannot move at all (when $n = 1$), the situation is absolutely safe. Notice that the climax vehicle density are nearly the same in traffic flow and safety index because both quantities are closely related to average velocity.

- **Average Energy Consumption**

Now we define another important parameter called the average energy consumption denoted E_0. We use it to measure the energy consumed during each acceleration. We approximate the quantity as proportionate to change of kinetic energy. For a particular vehicle n, we denote its velocity at time t and $t+1$ as $V^{(t)}(n)$ and $V^{(t+1)}(n)$. Furthermore, we make the assumption that each vehicle has unit mass. Now we can derive the formula for E_0:

$$E_0 \propto \frac{\sum_{t=1}^{T}\sum_{n=1}^{N}\frac{1}{2}m[(V^{(t)}(n))^2-(V^{(t-1)}(n))^2]}{Q}$$

$$\propto \frac{\sum_{t=1}^{T}\sum_{n=1}^{N}\frac{1}{2}m[V^{(t)}(n)-V^{(t-1)}(n)][V^{(t)}(n)+V^{(t-1)}(n)]}{Q}$$

$$\propto \frac{\sum_{t=1}^{T}\sum_{n=1}^{N}[V^{(t)}(n)+V^{(t-1)}(n)]}{Q}$$

In order for convenience, we evaluate the proportionate coefficient as 1. So that

$$E_0 = \frac{\sum_{t=1}^{T}\sum_{n=1}^{N}[V^{(t)}(n)+V^{(t-1)}(n)]}{Q}$$

We again plot the average energy consumption against vehicle density in **Figure 5**.

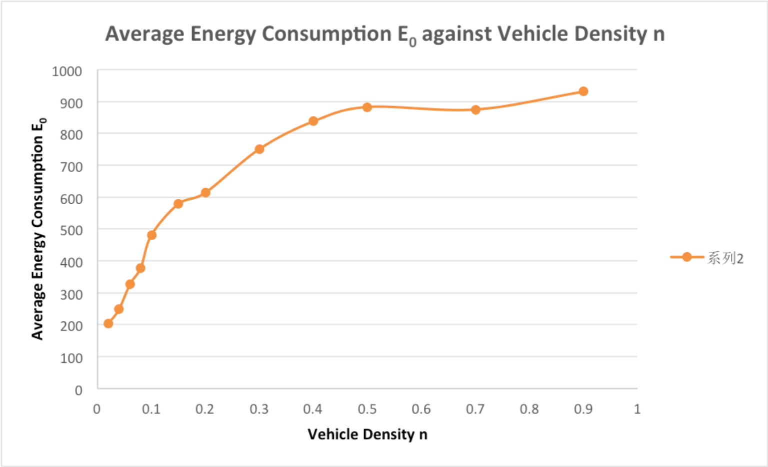

Figure 5. Relationship of Energy Consumption E_0 and Vehicle Density n (lower is better)

In the first half of the figure, the increase of E_0 could be explained by the frequent lane changing and acceleration due to a low vehicle density. In the latter half, although vehicles are much less likely to accelerate, notice that traffic flow also falls dramatically, therefore the average energy consumption still increases.

- **Optimal Maximum Velocity Estimation (Weighted)**

In this section, we will design a weighted function ϕ dependent on the three key variables---traffic flow Q, safety index α_{safe} and average energy consumption E_0. Then we will calculate an optimal speed limit to maximize the value of ϕ under light traffic and heavy traffic respectively.

We define the weighted function as follows:

$$\phi(Q,\alpha_{safe},E_0) = \omega_1\frac{Q}{1.2} + \omega_2(1-\frac{E_0}{800}) + \omega_3\sqrt[3]{1-(\frac{\alpha_{safe}}{0.4})^3}$$

$$s.t. \quad \omega_1 + \omega_2 + \omega_3 = 1$$

where $\omega_1, \omega_2, \omega_3$ are weight coefficient. We base on the following four rules to design our function:

- When ϕ reaches its peak value, we define the corresponding V_{max} to be the optimal maximum velocity;
- Since the three variables Q, α_{safe}, E_0 are independent of each other, ϕ should be expressed in the form of simple sums.
- Q and E_0 should appear in linear forms in ϕ;
- The value of ϕ should drop sharply when the safety index is rather high Hence α_{safe} should appear in the form of an upward-convex-function in ϕ.

Now under light traffic where $n = 0.1$ and heavy traffic where $n = 0.2$ respectively we can calculate the corresponding $V_{optimal}^{light}$ and $V_{optimal}^{heavy}$. We plot ϕ against different values of V_{max} under the condition $(\omega_1 = 0.5, \omega_2 = 0.2, \omega_3 = 0.3)$ in **Figure 6** and **Figure 7**.

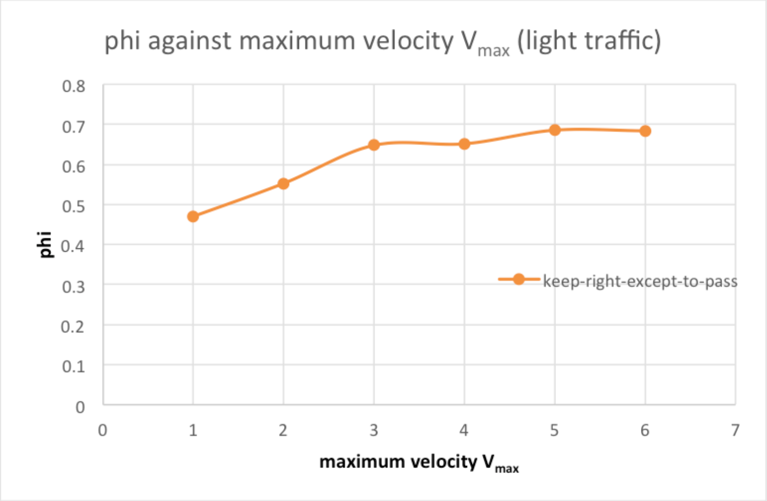

Figure 6. Optimal Maximum Velocity (Light Traffic)

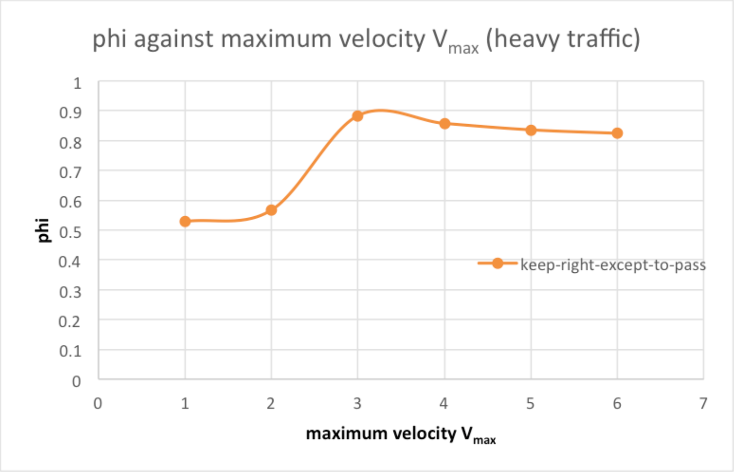

Figure 7. Optimal Maximum Velocity (Heavy Traffic)

Considering the duration of light traffic and heavy traffic in a given time period, we set the ratio 4:1 (light :heavy) to obtain the result for a "mixed- traffic" situation. See in **Figure 8.**

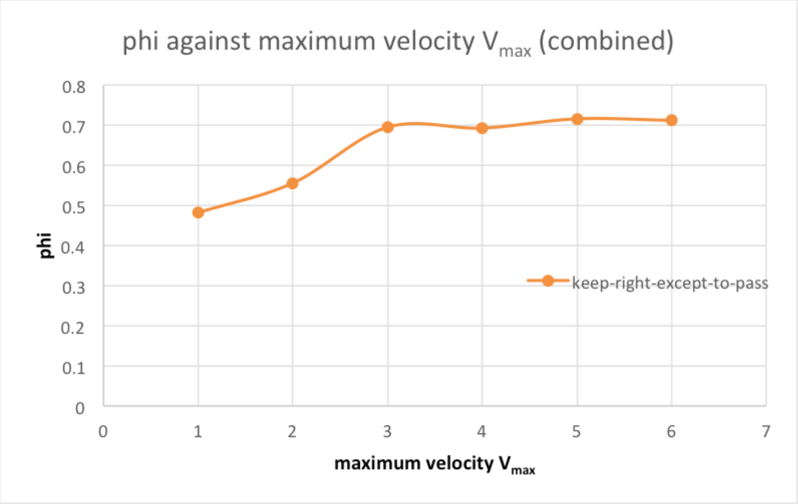

Figure 8. Optimal Maximum Velocity (combined)

As shown in **Figure 8**, the global maximum value of ϕ locates at $V_{max} = 5$, therefore the optimal maximum velocity $V_{optimal} = 5$. In other words, when we set the speed limit at 5, we achieve a weighted optimization of traffic flow, safety index and energy consumption.

- **Analysis of Under-Posted and Over-Posted Speed Limits**

In the section above, we have calculated the optimal maximum velocity $V_{optimal} = 5$. We will discuss the influence of under-posted and over-posted speed limits in light and heavy traffic respectively.

- **Light Traffic**

If a speed limit is over-posted, say, $V_{max} = 7$, traffic flow, safety index and average energy consumption all increase. In other words, we sacrifice energy and safety for larger traffic flow.

If a speed limit is under-posted, say, $V_{max} = 3$, traffic flow, safety index and average energy consumption all decrease. In other words, we sacrifice traffic flow for lower energy consumption and a safer condition.

- **Heavy Traffic**

If a speed limit is over-posted, say, $V_{max} = 7$, traffic flow will remain almost the same while energy consumption and safety index will rise. This is highly unwelcome.

If a speed limit is under-posted, say, $V_{max} = 3$, situation will be similar to that of light traffic. Still, we will be sacrificing traffic flow for lower energy consumption and a safer condition.

- **Visualization**

To make our simulation results more intuitive, we use MATLAB to visualize our simulation. In the figures below, each row signifies the distribution of vehicles with black dots representing vehicles occupation and margin representing empty space. Time increases from bottom to top and vehicles move from left to right. We visualize our simulation at low vehicle density ($n = 0.05$) and high density ($n = 0.2$) respectively in **Figure 9** and **Figure 10**. Subfigure on the left represents the left lane and the subfigure on the right represents the right lane.

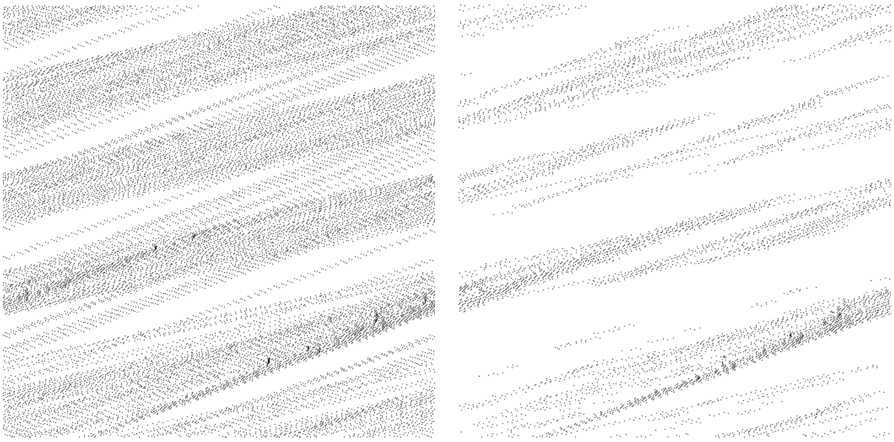

Figure 9. Visualization at Low Density

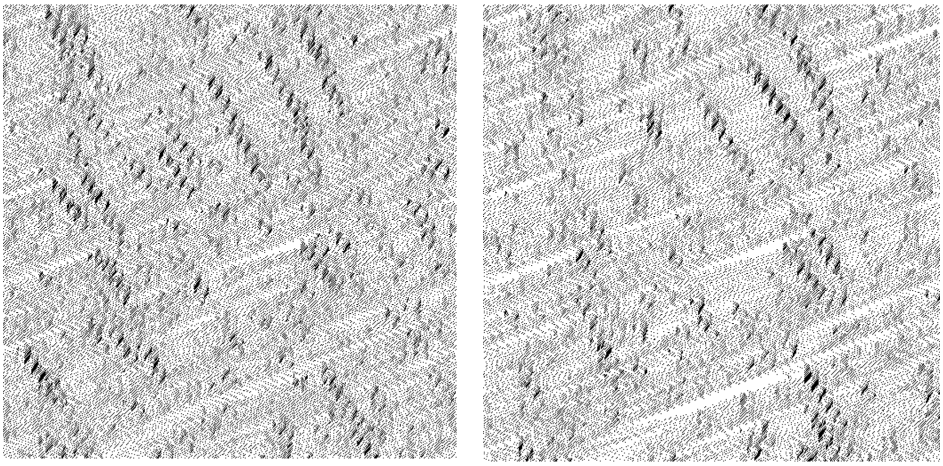

Figure 10. Visualization at High Density

3.2 Continuum Modeling for Right-Most Rule

Apart from the discrete, microscopic cellular automaton (CA) models we have adopted above, in this section we will establish a continuum, macroscopic model to describe the vehicle flow on the freeway.

3.2.1 Model Establishment

Our ultimate goal is to make a continuum model for the traffic flow on a double-lane freeway. We start our work from the single-lane situation. According to [4], we can use partial-derivative equations (PDE) system to describe the traffic flow on a single-lane freeway. There are two variables in the equation system: the average speed of vehicles $u(x,t)$ and the density of vehicles $\rho(x,t)$. The PDE system contains two parts:

- Continuity equation:

$$\frac{\partial u}{\partial t}+\frac{\partial(\rho u)}{\partial x}=0$$

- Dynamic equation:

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}=\frac{u_e(\rho)-u}{T}+c_0\frac{\partial u}{\partial x}$$

In the second equation, the right side contains a relaxation term $(u_e(\rho)-u)/T$, which reflects the process when a driver tries to adjust his velocity to the equilibrium velocity $u_e(\rho)$ in the time interval T and an anticipation term $c_0\partial u/\partial x$, which represents the process when the driver reacts to the traffic condition ahead with the propagation speed of small disturbance c_0.

Now we extend that model to accommodate the double-lane situation to accommodate the "keep-right-except-to-pass" rule. We will establish the equations for the two lanes separately. In order to describe lane-changing between the right (driving) lane and the left (overtaking) lane, we will introduce two variables S_{ji} and S_{ij}, which represent the lane changing rate. In particular, S_{ji} denotes the changing rate from lane i to lane $j (i, j \in \{1,2\})$. The overtaking lane is denoted by Lane 1 and the driving lane is denoted by Lane 2. Thus, the net lane change into lane i is $S_{ji} - S_{ij}$.

Hence the continuity equation could be written as:

$$\frac{\partial \rho_i}{\partial t} + \frac{\partial (\rho_i u_i)}{\partial x} = S_{ji} - S_{ij}$$

For the dynamic equation, we introduce two constant parameters r_1, r_2 to be the coefficients of S_{ji}, S_{ij}. The equation should be:

$$\frac{\partial u_i}{\partial t} + u_i \frac{\partial u_i}{\partial x} = \frac{u_{ei}(\rho_i) - u_i}{T_i} + c_{0i} \frac{\partial u_i}{\partial x} + r_1 S_{ij} - r_2 S_{ji} \qquad (i = 1,2)$$

According to [5], we assume:

$$S_{12} = a Q_{e1}(\bar{\rho}) \rho_1 (1 - \frac{\rho_2}{\rho_m})$$

$$S_{21} = a(1 + b(Q_{e1}(\bar{\rho}) - Q_{e2}(\bar{\rho}))) Q_{e2}(\bar{\rho}) \rho_2 (1 - \frac{\rho_1}{\rho_m})$$

where Q_{e1} denotes the equilibrium flow $Q_{e1} = \rho_1 u_{e1}(\rho_1)$
$\bar{\rho}$ denotes the average density of Lane 1 and Lane 2 $\bar{\rho} = (\rho_1 + \rho_2)/2$
ρ_m denotes the density in a jam
a, b are two constant parameters.

Such assumption is reasonable. Firstly, notice that the expression of S_{12} and S_{21} is not symmetric because of the asymmetricity of the double-lane freeway. Secondly, if Lane 2 is suffering from a traffic jam, then $\rho_2 = \rho_m$; if there are no vehicles in overtaking lane 1,then $\rho_1 = 0$. Under both circumstances, we can derive $S_{12} = 0$, which implies that no car can change into the driving lane when it has a traffic jam or when there are no vehicles on the overtaking lane.

3.2.2 Model Solution and Analysis

- **Study of Steady State**

According to theories of PDE, we can obtain a steady state solution of our model by setting the derivative terms to zero. Then we can derive the following two equations:

$$Q_{e1}(\bar{\rho})\rho_1(1-\frac{\rho_2}{\rho_m}) = (1+b(Q_{e1}(\bar{\rho})-Q_{e2}(\bar{\rho})))Q_{e2}(\bar{\rho})\rho_2(1-\frac{\rho_1}{\rho_m})$$

$$\frac{u_{ei}(\rho_i)-u_i}{T_i} + c_{0i}\frac{\partial u_i}{\partial x} + r_1 S_{ij} - r_2 S_{ji} = 0 \quad (i = 1{,}2)$$

Plugging in $2\bar{\rho} = \rho_1 + \rho_2$, we can derive the expression of $u_{e1}(\rho)$ and $u_{e2}(\rho)$ from the equations above according to [6]:

$$u_{e1}(\rho) = u_f(1-\frac{\rho}{\rho_m})/(1+E(\frac{\rho}{\rho_m})^4)$$

$$u_{e2}(\rho) = \begin{cases} u_f(1-\frac{\rho}{\rho_m})/(1+E(\frac{\rho}{\rho_m})^4)(\frac{1}{2}+\frac{\rho}{2\rho_c}) & (\rho < \rho_c) \\ u_f(1-\frac{\rho}{\rho_m})/(1+E(\frac{\rho}{\rho_m})^4) & (\rho \geq \rho_c) \end{cases}$$

where u_f denotes the free flow speed and E is a constant parameter.

Using the relationship given above, we can derive the relationship between u_i and $\bar{\rho}$. Then we use the definition $q_i = \rho_i u_i$ to express the flow on each lane with the average density $\bar{\rho}$.

According to [7], we assign the following values to the parameters:

$T_1 = T_2 = 10s, u_f = 30\,m/s, a = 0.01, b = 5, c_{01} = c_{02} = 10m/s$

$\rho_m = 0.14\,veh/m, r_1 = 150m^2/s, r_2 = 50m^2/s, \rho_c = 0.05\,veh/m, E = 100.$

Since we have established the traffic flow function of average density on both lane, we can plot the relationship as shown in **Figure 11.**

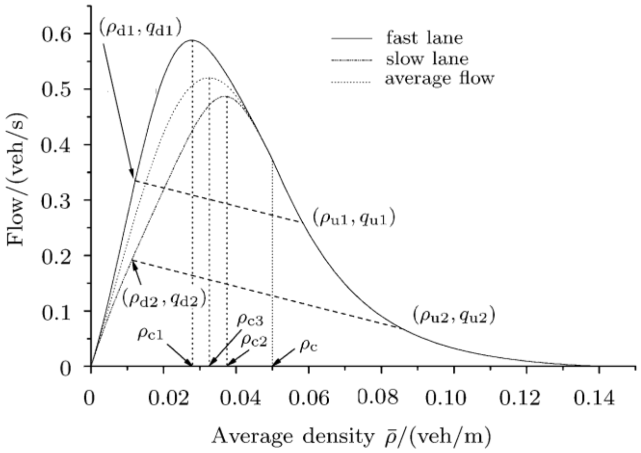

Figure 11. The Relationship of Traffic Flow Q and Average Density $\bar{\rho}$

In **Figure 11**, we can discover that all three kinds of flows experience a similar changing pattern---when the average density is relatively small, traffic flow experiences a monotonic increase as the density gets larger. After reaching its climax, the flow drops gradually to 0 as the density keeps rising. This changing pattern is identical to the results we have obtained in the discrete model. We can also discover that the overtaking (fast) lane reaches its climax sooner than that of the driving (slow) lane. Due to difference of the maximum velocity, the peak value of fast lane's traffic flow is larger than that of the slow lane. In addition, the curves of fast lane and slow lane almost overlap when $\rho > \rho_c = 0.05$. The overlapping part of curves actually indicates that the utility rate of both lanes is almost the same when ρ is relatively large.

- **Study of Numerical Simulation**

In this section, we mainly study the density function $\rho_i(x,t)(i = 1, 2)$ as a solution of the PDE system. In order to simulate the two-lane traffic movement, we assume that $L = 32.2\,km$. The following initial variation of the average density ρ_h proposed in [8] can be used on either the fast or the slow lane:

$$\rho(x,0) = \rho_h + \Delta\rho_h[\frac{1}{cosh^2(\frac{160}{L}(x - \frac{5L}{16}))} - \frac{1}{4cosh^2(\frac{40}{L}(x - \frac{11L}{32}))}]$$

Then, our model (the PDE system) can be solved by the numerical scheme given in reference [9]. Here we just provide the figures of the solution under three different conditions. (All of these three figures are dynamic traffic images of an asymmetric double-lane system with curve (a) representing the fast lane and curve (b) representing the slow lane.)

- **At Low Average Density $\bar{\rho}$**

According to **Figure 12**, any perturbations on the fast lane or the slow lane will quickly dissipate. Because of the lane-changing tendency, a cluster appeared in the fast lane may cause a small hump in slow lane.

On the opposite side, as shown in **Figure 13**, a cluster in slow lane may quickly dissipate because people trapped by the cluster in slow (driving) lane may choose the fast (overtaking) lane for passing.

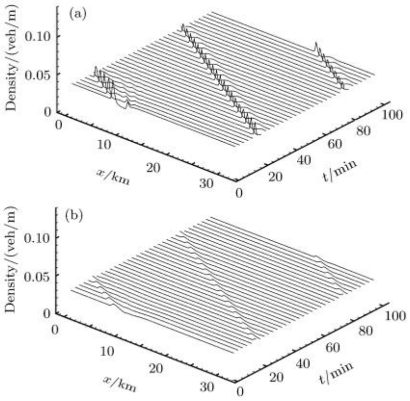

Figure 12. Low Density(Jam First Appeared on Fast Lane)

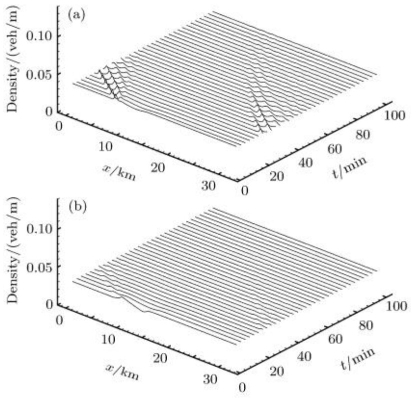

Figure 13. Low Density(Jam First Appeared on Slow Lane)

- **At Normal (or Higher) Average Density $\overline{\rho}$**

As average density continues to rise, free space for vehicles decreases and it becomes less likely for the double-lane system to maintain stability. Correspondingly, serious traffic jams and "stop-and-go" waves become more probable. See in **Figure 14**.

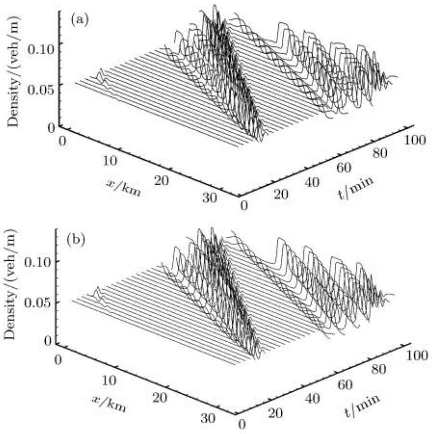

Figure 14. Normal or Higher Density

Parameter evaluation of the three figures:

Figure 12: $\rho_1 = 0.047 veh/m, \rho_2 = 0.021\ veh/m, \rho_h = 0.06\ veh/m$;
Figure 13: $\rho_1 = 0.047 veh/m, \rho_2 = 0.021\ veh/m, \rho_h = 0.06\ veh/m$;
Figure 14: $\rho_1 = 0.053 veh/m, \rho_2 = 0.053\ veh/m, \rho_h = 0.06\ veh/m$.

3.3 Comparison of Discrete Model and Continuum Model

In the discrete (CA) model, we focus on the relationship between traffic flow and vehicle density and obtain the following fact: when density is relatively small, flow rises as density increases; after reaching its peak value, flow then decreases gradually to 0. This coincides with the conclusion derived in the continuum model as shown in **Figure 9**. More intuitively, the visualization images of both models also convey the same conclusion.

3.4 Transferability Analysis of Model in Left-Handed Countries

According to our research, the difference between the left-most and right-most tradition is mainly historical. In addition, our model has no partiality for either left orientation or right orientation. In other words, orientation is symmetric in our model. Therefore our model could be transferred in left-handed countries with a single change of orientation.

4 Modeling for Alternative Freeway Traffic Rules

In this section, we will use similar *Monte Carlo* method to simulate the traffic flow under two alternative traffic rules: "slow-car-to-right" rule and "free driving & free overtaking" rule.

4.1 "Slow-Cars-To-Right" Rule

● Model Explanation

In the "slow-cars-to-right" rule, the slow cars are designated the right lane, the fast cars are designated among the left and right lanes and no lane changing is allowed. In other words, this is a model of two disconnected single lanes. We run two simulations. First, test a lane with only fast cars and calculate corresponding Q^f, α^f_{safe}, E^f_0; then test a lane with 60% fast cars and calculate corresponding Q^s, α^s_{safe}, E^s_0. The overall fast car percentage is 80% matching previous simulations. The final result is a mixture of previous results with weight (1:1).

$$Q = 0.5 \cdot Q^f + 0.5 \cdot Q^s$$
$$\alpha_{safe} = 0.5 \cdot \alpha^f_{safe} + 0.5 \cdot \alpha^s_{safe}$$
$$E_0 = 0.5 \cdot E^f_0 + 0.5 \cdot E^s_0$$

Parameter evaluation before the simulation is as follows:

The length of the lattices $L = 2048$;

The vehicle density $n = 0.02, 0.04, 0.06, 0.08, 0.1, 0.13, 0.16, 0.2, 0.25, 0.3$;

The number of the total vehicles $N = 2nL$;

The duration of the simulation $T = 4096$;

The deceleration probability $P_d = 0.1$;

The change lane probability $P_c = 0.7$;

The maximum velocity of fast car $V^f_{max} = 5$, maximum velocity for slow car $V^s_{max} = 3$.

- **Model Solution and Analysis**

Using computer simulation, we plot traffic flow Q, safety index α_{safe} and average energy consumption E_0 respectively against vehicle density n. The results will be shown in **Section 4.3**.

4.2 "Free Driving & Free Overtaking" Rule

- **Model Explanation**

Under the "free driving & free overtaking" rule, vehicles are free to drive and overtake with no specific requirement. The two lanes are symmetric and the only limitation is the respective maximum velocity V^f_{max} and V^s_{max} for fast vehicles and slow vehicles. Parameters and their evaluation are still the same as in the "slow-cars-to-right" rule.

- **Model Solution and Analysis**

Again, we consider the influence of this traffic through measuring of the three key factors---traffic flow Q, safety index α_{safe} and average energy consumption E_0. Further comparison of the three traffic rules will be shown in the **Section 4.3.**

4.3 Comparison of Different Traffic Rules

Same as that in the slow-cars-to-right rule, we will mainly focus on the first half of the figures to analyze the difference of the three rules.

- **Traffic Flow**

We plot traffic flow Q against vehicle density n in **Figure 15**.

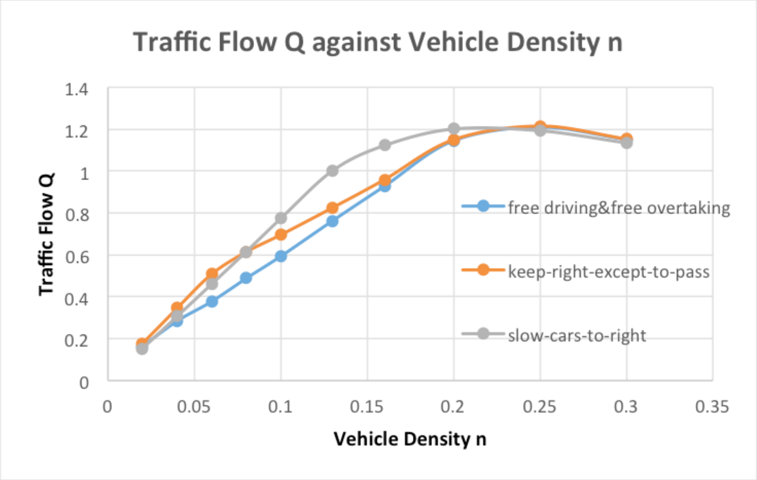

Figure 15. Traffic Flow under Different Traffic Rules

As shown in **Figure 15**, during light-traffic situation, traffic flow under the "keep-right-except-to-pass" rule and "slow-cars-to-right" rule are larger, traffic flow under the "free driving & free overtaking" is the relatively small.

During heavy-traffic situation, flow under three rules tends to be the same, which is easy to understand because the more probable jams make the advantages of the last two traffic rule hard to discern.

- **Safety Index**

We plot safety index α_{safe} against vehicle density n in **Figure 16**.

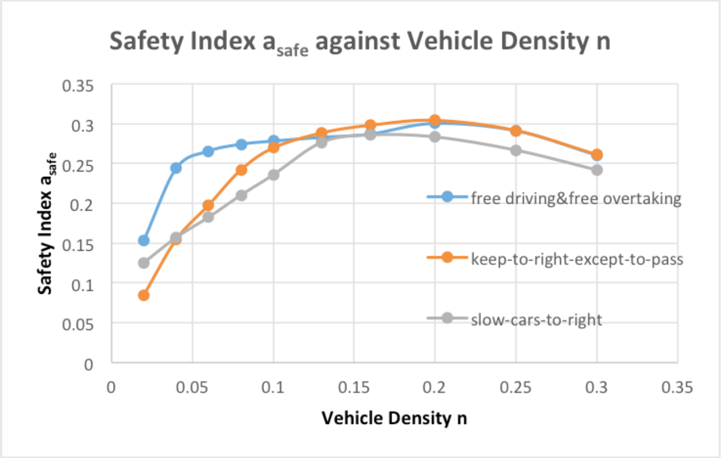

Figure 16. Safety Index under Different Traffic Rules (lower is safer)

During light-traffic situation, the condition under the "keep-right-except -to-pass" rule and "slow-cars-to-right" rule is much safer than that under the "free driving & free overtaking" rule.

During heavy-traffic situation, same as that of traffic flow, safety index under three traffic rules tend to be the same because vehicle tend to drive at a much lower speed which make the overall condition much safer.

● Average Energy Consumption

We plot average energy consumption E_0 against vehicle density n in **Figure 17**.

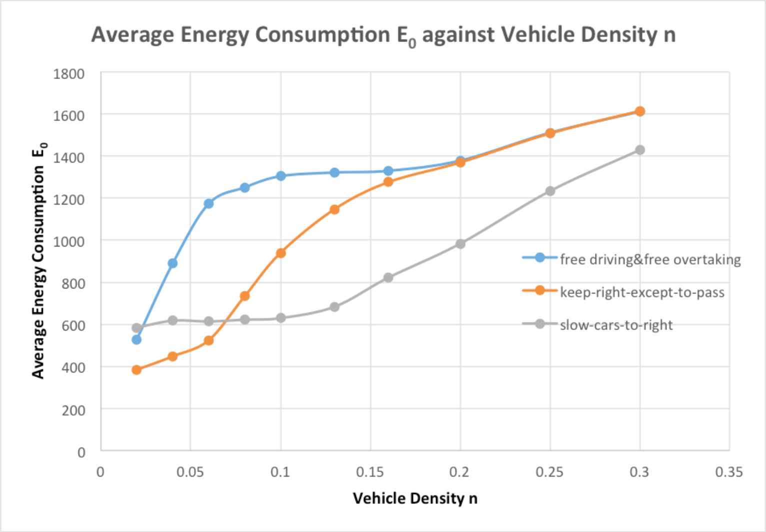

Figure 17. Average Energy Consumption under Different Traffic Rules (lower is better)

Since a large portion of energy consumption comes from traffic jams, and unstable lane changing easily result in traffic jams, the "free driving & free overtaking" rule will definitely be more energy-costly. Such trend is manifest during light traffic when acceleration is fairly easy.

During heavy traffic however, lane changing and acceleration become much more difficult, most of the energy consumption is from the accelerations and decelerations in traffic jams, therefore the average energy consumption under three rules tends to be the same.

● Summary & Suggestions

From the figures above, we can discover that "free driving & free overtaking" rule is inferior to the "slow-cars-to-right" rule and "keep-right-except-to-pass" rule. This accords with real life situation because most countries adopt a combinational policy of "slow-cars-to-right" and "keep-right-except-to-pass".

Meanwhile, it is unrealistic to adopt the "slow-cars-to-right" because total forbidding of lane changing and overtaking is unreasonable. Our suggestion is that maintain the policy of "keep-right-except-to-pass" rule and meanwhile construct separate fast and slow lanes for those freeways frequently suffered from car jams and obstructions.

5 Modeling for Intelligent-System Control

In this section, we make the general assumption that equipped with an intelligent system. The intelligent system knows about the exact velocity and location of all the vehicles on the freeway and controls them together. We aim to propose a new traffic rule such that the overall traffic flow of the freeway could be even larger.

5.1 "Median" Optimization

● Model Explanation

We now introduce a rule called “median” rule. The basic idea is to let vehicles with higher speed stay on the left lane more, while vehicles with lower speed stay on the right lane, reducing the possibility of fast cars following slow cars. The separation point is the median speed, which is known easily by a system that controls all the vehicles. The advantage of median speed is that it is automatically adjusted according to the current average speed, then exactly demand half of the cars go to the left lane and vice versa. Therefore this rule will not let the left lane and right lane become unbalanced both in light and heavy traffic comparing to the usage of a fixed speed cap.

The rule is as stated in **Figure 18.**

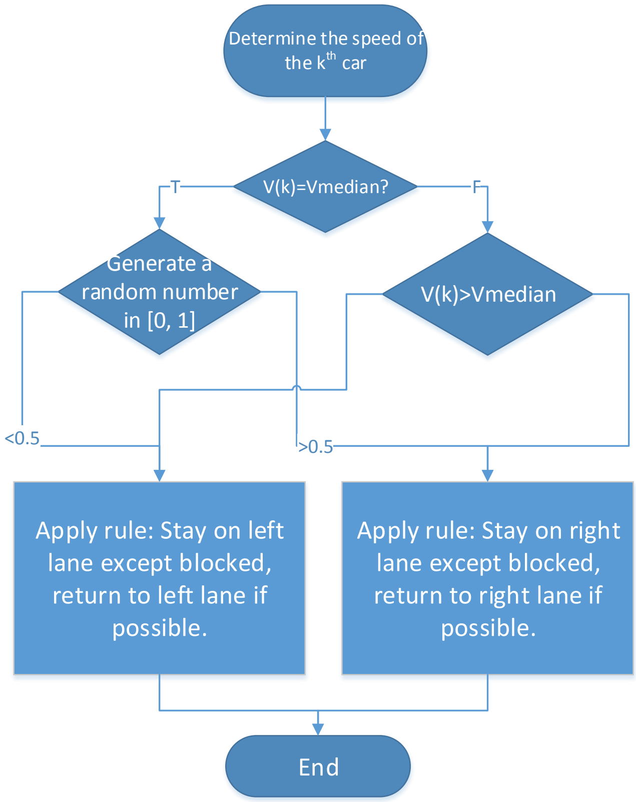
Determine the speed of the kth car
V(k)=Vmedian?
T
F
Generate a random number in [0, 1]
V(k)>Vmedian
<0.5
>0.5
Apply rule: Stay on left lane except blocked, return to left lane if possible.
Apply rule: Stay on right lane except blocked, return to right lane if possible.
End

Figure 18. "Median" Illustration

- ## Model Solution and Analysis

Under the same maximum velocity, we plot traffic flow, safety index and average energy consumption against different vehicle density. See in **Figure 19-21**.

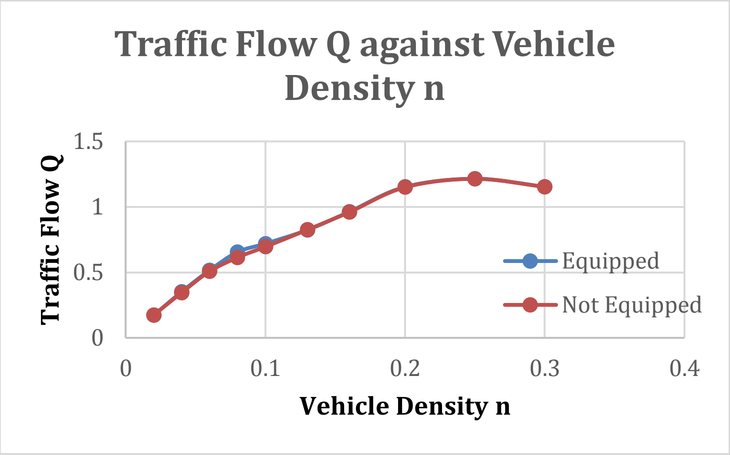

Figure 19. Relationship of Traffic Flow and Vehicle Density

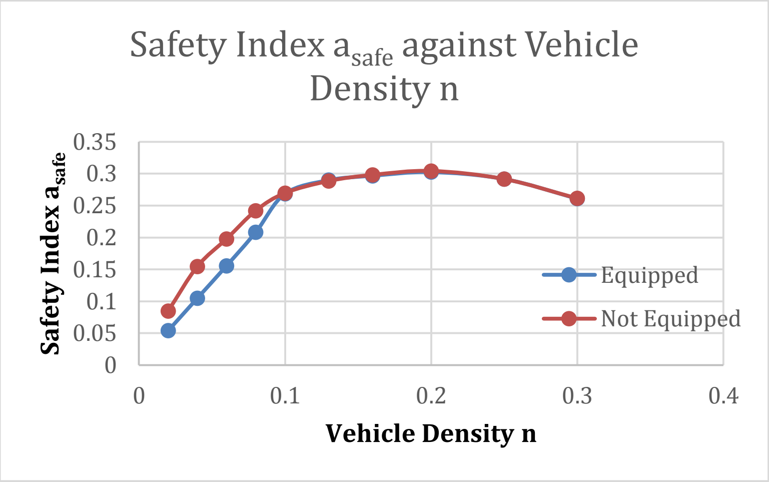

Figure 20. Relationship of Safety Index and Vehicle Density (lower is safer)

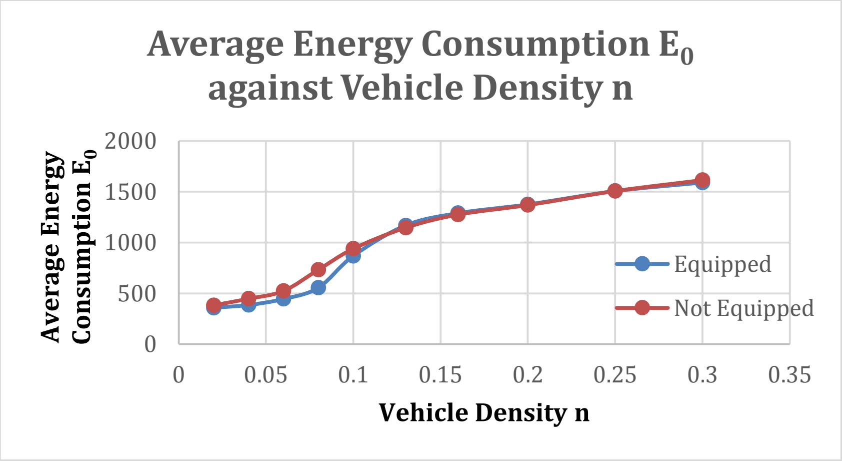

Figure 21. Relationship of Average Energy Consumption and Vehicle Density (lower is better)

From the above Figures, we can discover that the "median" optimization does perform better in terms of traffic flow, safety condition and energy consumption, even though such optimization is not very dramatic. This also indicates that the current "keep-right-except-to-pass" rule is already a pretty mature one.

5.2 Possible Further Improvements

The “median” rule is only a simple rule, more advantages of the intelligent systems can be utilized. In the discrete (CA) model, a driver can only change his speed according to the distance of cars closely in front or behind because he doesn't know about the location of other cars and intentions of their drivers. Therefore when he is in a car flow without any gaps as shown in **Figure 22**, the only thing he can do is stop and wait for the front car to move. That's why we only compare his current velocity V with ΔX_p^f and ΔX_o^f.

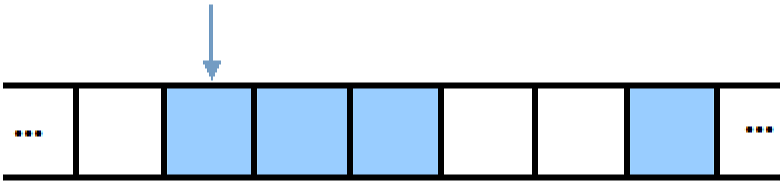

Figure 22. Illustration of Online Algorithm

But with an intelligent system, it controls all the cars at the same time. Under such circumstances, it can optimize traffic flow by accelerating the three cars together with the same acceleration. This approach is extremely effective in traffic jams.

6 Superiority and Weakness

6.1 Superiority Analysis

- By reasonable discretization and *Monte Carlo* method, CA model properly simulates normal driving and overtaking on real-life double-lane freeways;
- The solution of the CA model could be presented via various forms and standards, visualization technique makes the result even more intuitive;

- Our model has robust flexibility, only minor modification could realize simulation under different traffic rules and parameters;
- By comparing with other traffic rules, we can fully realize the influence of keep-right-except-to-pass rule from various aspects;
- By designing algorithms under an intelligent system, our model has reference value in terms of future freeway construction;
- Our continuum model assigns realistic values to parameters, confirming the results derived from the discrete model from a macroscopic perspective.

6.2 Weakness Analysis

- Due to the restriction of computer simulation, some of our assumptions are highly ideal, such as the length of vehicles and the value of acceleration.
- The core of our system depends on computer simulation, therefore the calculation of key quantities such as traffic flow and safety index is time-costly.

6.3 Future Research

6.3.1 Possible Optimization of Discrete Model

- If technique permits, we strive to make computer simulation as similar as possible to realistic situations. One possible measures is that we can use 1 minute as the basic time unit and assign realistic values to lattice length (such as 0.5km). Since most vehicles travel at velocity between 60 km/h and 120 km/h, within one time unit (1 min), a vehicle is expected to pass 2-4 lattices within one minute.
- For more accurate approximation, vehicle profile should be taken into consideration. For example, limousines, sedans, jeeps are usually small but relatively fast, we can assign them with one-lattice length and larger maximum velocity; trucks, on the other hand, could be assigned with length of multiple lattices and a smaller maximum velocity.
- Finally, we add a more professional factor in the transportation field---stopping sight distance and its limit. The stopping sight distance is used to measure several secure distances in driving. We can assign an integer to the stopping sight distance, denoted as d_0. When the front gap is larger than the stopping sight distance ($\Delta X_P^f > d_0$), we compare $V+1$ with

d_0 instead of with ΔX_P^f or ΔX_o^f to determine the vehicle's velocity at next time interval. We can further combine weather conditions to evaluate the stopping sight distance. Such model is of great importance in deciding whether to close freeways under severe weather conditions.

6.3.2 Possible Optimization of Continuum Model

- In many countries around the world, the number of freeways are dramatically increasing. We can continue to develop multi-lane(more than two lanes) models. Speed limits could also be incorporated in the continuum model.
- Construct a systematic theory to analyze the graphic form of $\rho(x,t)$ and look for accurate analytical expressions.

7 Conclusion

We use a discrete model and a continuum model research upon the influence of "Keep-Right-Except-To-Pass" rule on traffic flow, safety index and average energy consumption respectively. Furthermore, to fully determine the pros and cons of the rule, we continue to build models to analyze two alternative traffic rules---the "Slow-Cars-To-Right" rule and the "Free Driving & Free Overtaking" rule. Finally, we propose two possible rules to optimize traffic flow when equipped with an intelligent system. The results of our models are as follows:

- **“Keep-Right-Except-to-Pass” Rule**

According to our model, traffic flow and safety index first rise to peak value then decrease to zero with increasing vehicle density, while energy flow keeps rising but with a bating speed. Furthermore, under-posted and over-posted speed limits relative to our calculated optimal maximum velocity have different influences during light and heavy traffic. In addition, our model could be transferred in left-handed countries with a single change of orientation.

- **Alternative Traffic Rules**

The "Keep-Right-Except-To-Pass" rule and the "Slow-Cars-To-Right" rule are superior (to the "Free Driving & Free Overtaking" rule), a combination of these two rules will have strengthening effects especially on heavy-traffic freeways.

- **Intelligent System Control**

An ideal online algorithm is discussed, and a "median" optimization method is simulated and proves to be beneficial to enlarge traffic flow.

8 References

[1] Peilin Li, "*Globalization and the 'Left-Most' Policy of Automobiles*", Journal of Financial Introduction, Vol.8, 2004.

[2] Chowdhury, D., Wolf, D. E., & Schreckenberg, M. (1997). Particle hopping models for two-lane traffic with two kinds of vehicles: Effects of lane-changing rules. *Physica A: Statistical Mechanics and its Applications*, 235 (3), 417-439.

[3] Giordano, Frank R. A first course in mathematical modeling. Cengage Learning, 2013.

[4] Haijun Huang · Tieqiao Tang · Ziyou Gao:Continuum modeling for two-lane traffic flow, *Acta Mech Sinica* (2006) 22: 131–137.

[5] Tang Chang-Fu, Jiang Rui, Wu Qing-Song:Extended speed gradient model for traffic flow on two-lane freeways. *Chinese Physics*, 2007/16(06)

[6] Lee H Y, Lee H W and Kim D 1999 *Phys. Rev.* E 59 5101

[7] Nagel K and Schreckenberg M 1992 J. Phys. I 2 2221

[8] Kerner B S and Konsh¨auser P 1993 *Phys. Rev.* E 48 2335

[9] Fu C J, Wang B H, Yin C Y and Gao K 2006 *Acta Phys.Sin.* 55 4032

9 Appendix

- C++ Source Code for Monte Carlo Simulation:

gen.h:

```
#ifndef __GEN
    #define __GEN


#include <iostream>
#include <cstdio>
#include <ctime>
#include <cstdlib>
#include <vector>
#include <cassert>
using namespace std;
/*
#define RATIO 0.3
#define L 512
#define VMAX 7
```

```
#define VBASE 2
*/
int random(int low, int high){
	return (rand()%(high-low+1)+low);
}
void gen(const char *fpath, int L, double den, int Vmax0, int Vmax1, double Pv){
	FILE *fp = fopen(fpath, "w");
	int n=den*L;
	vector<int> a;
	for (int i = 0; i < L && a.size() < n; i ++)
		if (R((n-a.size())*1.0/(L-i)) || a.size()+L-i <= n)
			a.push_back(i);
	assert(a.size() == n);
	fprintf(fp, "%d, %d\n", L, n);
	int v;
	for (int i = 0; i < n; i ++){
		v=R(Pv)?Vmax0:Vmax1;
		fprintf(fp, "%d, %d, %d\n", a[i], v, random(v/2, v));
	}
	fclose(fp);
}
#endif // __GEN
```

SaveBMP.h:

```
#ifndef __BMP
	#define __BMP
FILE *SaveBmp(const char* filename,int height,int width){
	unsigned int w = ((width+7)/8+3)/4*4;
	unsigned int size = w*height+0x3E;
	unsigned short head[]={
		0x4D42,size&0xffff,size>>16,0,0,0x3E,0,0x28,
		0,width&0xffff,width>>16,height&0xffff,height>>16,1,1,0,
		0,0,0,0,0,0,0,0,0,0,0,0,0,0xffff, 0x00ff
	};
	FILE* fp=fopen(filename,"wb");
	if(!fp)return NULL;
	fwrite(head,1,sizeof(head),fp);
	return fp;
}
#endif
```

lane.h:

```
#ifndef __LANE
```

```
    #define __LANE

#include <iostream>
#include <cstdio>
#include <vector>
#include <cmath>
#include <cassert>
#include <cstdlib>
#include <ctime>
#include <algorithm>
#include <cstring>

#define out(x) cout<<(#x)<<"="<<x<<endl;

using namespace std;

int L, T = 2048, VISUALIZE = 0, TOT = 5;
double Pd = 0.2, Pc = 0.6, Pv = 0.8;
int Voff = 2, RULE = 0, Vmax0 = 4, Vmax1 = 3;
double ECost = 0;
double AlphaSafety=0;

bool R(const double &P){
    return rand()<RAND_MAX*P;
}
struct car{
    int x, v, vmax;
    car(int x, int v, int vmax):x(x), v(v), vmax(vmax){}
    car(){}
    bool operator<(const car &rhs) const {
        return x < rhs.x;
    }
};
#define fn(i) ((i)>=n?(i)-n:((i)<0?(i)+n:(i)))
#define fm(i) ((i)>=m?(i)-m:((i)<0?(i)+m:(i)))
#define fL(x) ((x)>=L?(x)-L:((x)<0?(x)+L:(x)))
#define takemin(x, y) x=(x)<(y)?(x):(y)
unsigned char buf[10000];
struct lane{
    vector<car> c;
    int cnt[7];
    int tot() const {
        int tot=0;
        for(int i=0; i<7; i++)
```

```
            tot+=cnt[i];
        return tot;
    }
    car &operator[](int i){return c[i];}
    lane(const char *fpath){
        memset(cnt, 0, sizeof cnt);
        FILE *fp = fopen(fpath, "r");
        int n, t;
        fscanf(fp, "%d, %d, %d ", &L, &n, &t);
        c.resize(n);
        for(int i=0; i<n; i++){
            fscanf(fp, "%d, %d, %d ", &c[i].x, &c[i].vmax, &c[i].v);
            assert(c[i].vmax >= c[i].v);
            cnt[c[i].v]++;
        }
        fclose(fp);
    }
    lane(){}
    bool OK() const {
        for(int i=0; i<c.size(); i++)
            if(c[i].x<0 || c[i].x>=L)return false;
        for(int i=0; i+1<c.size(); i++)
            if(c[i].x >= c[i+1].x)return false;
        if (tot() != c.size()) return false;
        return true;
    }
    int size() const {
        return c.size();
    }
    int advance(vector<car> &t){
        int n;
        t.resize(n = c.size());
        if(n==0)return 0;
        for(int i=0; i<c.size(); i++){
            int j=fn(i+1);
            t[i]=c[i];
            if (t[i].v < t[i].vmax) t[i].v++;
            int DXf = fL(c[j].x-c[i].x-1);
            if (t[i].v > DXf) t[i].v = DXf;
            if (t[i].v > 0 && R(Pd)) t[i].v--;
            t[i].x = fL(t[i].x + t[i].v);
            if (t[i].v > c[i].v) ECost += t[i].v+c[i].v;
        }
        if(t[n-1].x<t[0].x){
            for(int i=1; i<n; i++)
                c[i]=t[i-1];
            c[0]=t[n-1];
            return 1;
        } else swap(t, c);
        memset(cnt, 0, sizeof cnt);
        for(int i=0; i<c.size(); i++)
            cnt[c[i].v]++;
        return 0;
    }
    void print(){
        for(int i=0; i<c.size(); i++)
            printf("(%d, %d) ", c[i].x, c[i].v);
        puts("");
    }
    void pos(FILE *fp){
        int ll = ((L+7)/8+3)/4*4*8;
        unsigned char tmp = 0;
        for(int i=0, j=0; i<ll; i++){
            int b=j<c.size()?(i<c[j].x?1:(i!=c[j++].x)):1;
            tmp = (b<<7)|(tmp>>1);
            if(i&0xff)buf[i>>3]=tmp;
        }
        fwrite(buf, 1, ll/8, fp);
    }
};
#endif
```

CA.cpp:

```
#include <string>
#include "lane.h"
#include "gen.h"
#include "SaveBMP.h"
lane l[2];
int cnt[2];
char VPATH[200];
bool rule_s(lane &l0, lane &l1, int i, int j, int DXf)
{
    int m = l1.size();
    if (m==0) return DXf > 1;
    int k = fm(j-1);
    if(DXf > 1 && l1[j].x != l0[i].x && //! no parralel car
       fL(l0[i].x - l1[k].x) > min(l1[k].v+1, l1[k].vmax)) //! no accident in the other lane
       return 1;
    return 0;
}
bool rule0(lane &l0, lane &l1, int i, int j, int DXf)
{
    int m = l1.size();
    if (m==0) return DXf < l0[i].vmax;
    if(DXf < l0[i].vmax){ //! Not enough space in this lane
        int k = fm(j-1);
        if(fL(l1[j].x - l0[i].x) > DXf)      //! More space in the other lane
            if (R(Pc)) return 1;
    }
    return 0;
}
bool rule1(lane &l0, lane &l1, int i, int j, int DXf)
{
    if (l1.size() == 0) return R(Pc);
    return R(Pc) && (fL(l1[j].x - l0[i].x)> min(l0[i].v+1, DXf));
    //! Enough or more space in the other lane
}
int median(int *c0, int *c1){
    int cnt[7];
    for(int i=0; i<7; i++)
        cnt[i] = c0[i] + c1[i];
    int tot = 0;
    for(int i=0; i<7; i++)
        tot += cnt[i];
    tot /= 2;
    for(int i=0; tot>0 && i<7; i++){
        tot -= cnt[i];
        if (tot<=0) return i;
    }
    return 6;
}
void chLaneSim(lane &l0, lane &l1, vector<car> &t0, vector<car> &t1, int dir){
    int n=l0.size(), m=l1.size(), j=0;
    t0.clear(); t1.clear();
    bool flag = 1;
    if(!n){
        swap(t0, l0.c);
        return;
    }
    for(int i=0; i<n; i++){
        int DXf = fL(l0[fn(i+1)].x - l0[i].x);
        bool chLane = 1;
        for(; flag && j<m && l1[j].x<l0[i].x; j++);
        if (j==m) flag=j=0;
        chLane = chLane && rule_s(l0, l1, i, j, DXf);
        if (RULE == 16) chLane = 0;
        int rule, Vmedian = median(l0.cnt, l1.cnt);
        bool ff = l0[i].v>Vmedian ? 1 : (l0[i].v==Vmedian?R(0.5):0);
        rule = (dir<<1) + ((RULE&32) ? ff : (l0[i].vmax==Vmax0));
                                          //! 0: l->r slow, 1: l->r fast
        rule = (RULE>>rule) & 1;         //! 2: r->l slow, 3: r->l fast
        switch(rule){                    //! two_car: 1001=9, sym: 0000, sym_two_lane: 1100=12
            case 0: chLane = chLane && rule0(l0, l1, i, j, DXf); break;
            case 1: chLane = chLane && rule1(l0, l1, i, j, DXf); break;
            default: break;
        }
        if(chLane)t1.push_back(l0[i]);
        else t0.push_back(l0[i]);
    }
}
void sim(lane &l0, lane &l1){
    vector<car> t00, t01, t10, t11;
    chLaneSim(l0, l1, t00, t01, 0);
    chLaneSim(l1, l0, t11, t10, 1);
    vector<car> t0, t1;
    t0.resize(t00.size() + t10.size());
    t1.resize(t01.size() + t11.size());
    merge(t00.begin(), t00.end(), t10.begin(), t10.end(), t0.begin());
    merge(t01.begin(), t01.end(), t11.begin(), t11.end(), t1.begin());
    swap(t0, l0.c);
    swap(t1, l1.c);
    cnt[0] += l0.advance(t0);
    cnt[1] += l1.advance(t1);
}
double func(int dx, int v){
    if(v==0)return 0;
    else return exp(-dx/v);
}
double cal_safety(lane &l){
    double res=0;
    if(l.size()==0)return 0;
    for(int i=0; i+1<l.size(); i++)
        res += func(l[i+1].x-l[i].x, l[i].v);
    res += func(l[0].x-l.c.back().x+L, l.c.back().v);
    return res;
}
```

```
void gao()
{
    l[0] = lane("lane0.csv");
    l[1] = lane("lane1.csv");
    FILE *fout0, *fout1;
    if(VISUALIZE){
        fout0 = SaveBmp((string(VPATH)+"0.bmp").c_str(), T, L);
        fout1 = SaveBmp((string(VPATH)+"1.bmp").c_str(), T, L);
    }
    int t;
    if(VISUALIZE) t=0; else t=T/5;
    while (t--) sim(l[0], l[1]);
    t = T;
    cnt[0] = cnt[1] = 0; AlphaSafety = 0; ECost = 0;
    while (t--) {
        sim(l[0], l[1]);
        l[0].OK();
        l[1].OK();
        if(VISUALIZE){
            l[0].pos(fout0);
            l[1].pos(fout1);
        }
        AlphaSafety+=cal_safety(l[0])+cal_safety(l[1]);
    }
    AlphaSafety/=l[0].size()+l[1].size();
    AlphaSafety/=T;
    //printf("Q=%.5lf AlphaS=%.5lf\n", cnt/2.0/T, AlphaSafety);
    if(VISUALIZE){
        fclose(fout0);
        fclose(fout1);
    }
}
pair<double, double> stat(vector<double> &a){
    assert(a.size()>=2);
    double mean=0;
    for(int i=0; i<a.size(); i++)
        mean+=a[i];
    mean/=a.size();
    double s=0;
    for(int i=0; i<a.size(); i++)
        s+=(a[i]-mean)*(a[i]-mean);
    s/=a.size()-1;
    s=sqrt(s);
    return make_pair(mean, s);
}
int main()
{
    srand(time(0));
    FILE *fin = fopen("coef.prn", "r");
    FILE *fout = fopen("log.csv", "a");
    if (fin == NULL || fout == NULL) return -1;
    int sum=0;
    double den=0;
    fgets((char *)buf, 10000, fin);
    while(1){
        if(fscanf(fin, "%d %d %lf %lf %lf %lf %d %d %d %d %s",
            &L, &T, &den, &Pd, &Pc, &Pv, &Vmax0, &Vmax1, &VISUALIZE, &RULE, VPATH)!=11){
            break;
        }
        fprintf(fout, "%d, %d, %lf, %lf, %lf, %lf, %d, %d, %d, ",
                        L,  T,  den, Pd,  Pc, Pv, Vmax0, Vmax1, RULE);
        vector<double> q[2], AlphaS, EC;
        int REP = VISUALIZE?2:20;
        for(int i=0; i<REP; i++){
            gen("lane0.csv", L, den, Vmax0, Vmax1, Pv);
            gen("lane1.csv", L, den, Vmax0, Vmax1, Pv);
            gao();
            q[0].push_back(cnt[0]*1.0/T);
            q[1].push_back(cnt[1]*1.0/T);
            AlphaS.push_back(AlphaSafety);
            ECost /= cnt[0]+cnt[1];
            EC.push_back(ECost);
            printf("%d/%d RULE=%d Den=%.3lf Q0=%.3lf Q1=%.3lf AS=%.3lf EC=%.3lf\n", i+1, REP, RULE, den,
q[0].back(), q[1].back(), AlphaS.back(), EC.back());
        }
        pair<double, double> t0 = stat(q[0]), t1 = stat(q[1]), t2 = stat(AlphaS), t3 = stat(EC);
        fprintf(fout, "%.5lf, %.5lf, %.5lf, %.5lf, %.5lf, %.5lf, %.5lf, %.5lf\n", t0.first, t0.second, t1.first, t1.second, t2.first,
t2.second, t3.first, t3.second);
    }
    fclose(fout);
    fclose(fin);
}
```