1 A brief description of the algorithm

1.1 basic assumption and parameters

The platform considers the maximum acceleration of the vehicle is a_{max}, which is generally used in the acceleration process and tbe braking process. Also, the maximum speed limit is set to v_{max}, while the minimum speed limit is set to v_{min}.

No matter which strategy is chosen, if the acceleration given by the strategy is greater than the maximum acceleration a_{max}, or the calculated speed is greater than the maximum speed or less than the minimum speed, the program will automatically set these values to the boundary value. When the vehicle has to stop because of the red light, the minimum speed of the vehicle is set to 0, describing the parking behavior which may exist, however, the behavior of reversing is never allowed.

Also, we assume the expected speed of the vehicle is set to 80% of the maximum speed limit.

In order to simpliy the simulation process, we assume that during the process of driving, there is no lane changing behavior. This hypothesis is also acceptable in the real life, because of the restrictions imposed by driving habits and traffic laws & regulations, the vehicle should finish changing lane before enter the traffic junction. Therefore this assumption is reasonable. What's more, just from the perspective of traffic efficiency, lane-changing can only reduce the average efficiency of the traffic, therefore, as the primary inspection of the platform is efficiency, lane-changing situation is not taken into consideration.

Therefore, the car driving in the lane can be described as a queue, e.g. Q_{in}

1.2 The Generation of the vehicles

It is assumed that the arrival process of the vehicle is approximately a Poisson process, which is limited and adjusted by the following algorithm

Assuming that the intensity of the Poisson process isλ, the arrival time interval of the vehicle S_n, S_{n-1}can be expressed as

$$P((S_n - S_{n-1} \leq t) = 1 - e^{-\lambda t}$$

Because the simulation time scale is set to 0.1s, the average hourly traffic flow is

$$\bar{F} = 36000\lambda$$

For the traffic flow in one direction, after the average hourly traffic flow is set by the traffic flow adjustment slider described above, The vehicles in this direction will be generated according to the Poisson process with a specified intensity of $\lambda = \dfrac{\bar{F}}{3600}$, and will be pushed into the pending queue $Q_{pending}$in each direction.

After generating the vehicle, the next step is to consider which lane the vehicle belongs to, and it is assumed that the possibility of the vehicle appearing in each safe lane is same. Firstly, the condition to determine a safe lane is that the distance of last vehicle in the lane and line which the vehicles are generated (S_{start}) is greater than the safe distance S_{safe}, that is,

$$X_{-1} - S_{start} \leq S_{safe}$$

Screening all lanes that meet the situation above in this direction, and place the vehicle in $Q_{pending}$ into one of these lanes randomly, while the position of the vehicle newly generated is set to S_{start}.

If there is no lanes suitable for placing the vehicle currently, the vehicles will remain in $Q_{pending}$ and wait for the next simulation session.

The generation of the vehicle is done in the second step of the simulation cycle, just following the process control model.

1.3 The formation and evacuation of the queue in the intersection

To describe the phenomenon of queuing aat the intersection, a queue Q_{block} is introduced to each lane towards the intersection. And according to the current signal phase and the length of the queue, all of the control strategies mentioned take these simple algorithm below.

1. Red Light

   If the distance between the first vehicle in Q_{in} and the last vehicle in Q_{block} is less than the control range $S_{control}$, the first vehicle in Q_{in} brake and is expected to stop just behind the last vehicle in Q_{block}. The expected distance between these two vehicles is defined as S_{stop}, describing the distance between the vehicles in the Q_{block}.

   If the Q_{block} is empty and the distance between the first vehicle in Q_{in} and the stopping line is less than $S_{control}$, the first vehicle in Q_{in} brake and is expected to stop just on the stopping lineS_{end}.

   If the first vehicle in the $Q_{in}$is too far that neither of the two above are satisfied, the first vehicle in the Q_{in} is supposed to drive freely.

   Also, the vehicles in the Q_{block} should stop and wait until the light truns green.

2. Green Light distance between the first vehicle in Q_{in} and the last vehicle in Q_{block} (if it exists) should not be less than S_{safe}. And a braking is taken if the distance mentioned above is too short.

Therefore, the brief idea of the algorithm above can be described as the fake code below

```
01 if (Q_block.empty()){
02  if (Light == Green)
03   Q_in.first().drive_freely();
04  else
05   Q_in.first().brake_to(S_end);
06 }
07 else{
08  if(Q_block.last().pos-Q_in.first().pos< S_control)
09   Q_in.first().brake_to
10   (Q_block.last().pos-S_stop);
11  else
12   Q_in.first().drive_freely();
```

```
13 }
```

Where the brake_to(desired_pos) method means braking to a desired position.

Meanwhile, the growth and the dissipation of the queue Q_{block} follow the method below.

1. Increase

   If the first vehicle in Q_{in} is closer than S_{stop} from the last vehicle in Q_{block} (it is set to S_{end} if the Q_{block} is empty), it is removed from the Q_{in} and pushed into the Q_{block}

2. Dissipation

   When the traffic light is green, all of the vehicles in the Q_{block} is moving forward at the speed of v_{dis}, since the speed is relatively low, the acceleration process and any other phenomenon can be ignored. When the vehicle moves over the stopping lane and enters the intersection, it is removed from the Q_{block}

And the method above can be described as the fake code below

```
01 if (light == green){
02  Q_block.all_move_forward(v_dis)
03 }
04 else{
05  if(!Q_block.empty()){
06   if(Q_block.last().pos-Q_in.first().pos
07    < S_stop){
08    Vehicle v;
09    v=Q_in.getfirst();
10    Q_block.push(v);
11   }
12  }
13  else{
14   if(S_end-Q_in.first().pos< S_stop){
15    Vehicle v;
16    v=Q_in.getfirst();
17    Q_block.push(v);
18   }
19  }
20
21 }
```

All above describe the formation and evacuation of the queue in the intersection and its interaction with other models. The differences caused by the assumption can be ignored. Thanks to this model, it is easy for us to decouple the driving model and the queuing model, which makes it much easier to implement the other algorithm

1.4 Basic models

In the description of manual driving and automatic driving vehicle models, the car-following model and free driving model are frequently used models. Therefore, we will describe these models before discussing the control strategies.

1.4.1 The implementation of the car-following method

HERE IS THE BRIEF INTRO OF FOLLOWING MODEL

1.4.2 The implementation of the free-driving method

1.5 Manual driving model

The plantform provides three models to verify the effectiveness of the platform, and the first model is manual driving model.

As the name of the model indicates, manual driving model is used to describe the behavior of a vehicle driving by human. This model is often used to provide a 'basic' traffic efficiency, while other control strategies are expected to behave better than this simple model.

Firstly, vehicles which are far (greater than S_{inter}) from the intersection are taken into consideration. Because they are so far from the intersection, they do not consider the influence of the traffic light.

Therefore, these vehicles mat take the car-following method or the free-driving method. If a vehicle is the first one in Q_{in}, it has to take the the free-driving method.

As for the ones which are closer to the intersection, different method will be chosen to meet the different situation below.

1. Green traffic light with an empty waiting queue Q_{block}

   In this situation, the behavior of the vehicles is similar to the behavior described above.

   In addition to the behavior mentioned, the remain time of the green light is calculated, if the remaining time is less than T_{safe}, the possibility of entering the intersection during the green light may be taken into consider, if it is difficult for the vehicle to pass the stopping line before the light turns red, the vehicle may behave like the traffic light is red.

2. Green traffic light while the waiting queue Q_{block} is not empty

   During this situation, the vehicles discussed may brake or accelerate to set its speed to v_{dis}, and try to enter the queue Q_{block}, once it enters the Q_{block}, it will move with the other vehicles in the queue.

   Also, if the remaining time is less than T_{safe}, the program may consider slow down the vehicle and prepare to stop.

3. Red traffic light

   In this case, the vehicles discussed will behave just like what we have discussed in 1.3

What's more, we focus on portraying the behavior of the first vehicle in the driving queue Q_{in}, since the vehicles behind it can be controled easily by car-following method.

Taking all of the situations into consideration. the manual driving model can be abstract as the fake code below.

```
01 //the code below just consider the vehicle v
02 if(S_end-v.pos>S_inter)//far from intersection
03 {
04  //code block 1
05  if(car-following_requirement_met())
06   //the car-following requirement is   met
07   car_following();
08  else
09   //the head car of the queue or
10   //the car-following requirement isn't met
11   free_driving();
12 }
13 else //close to the intersection
14 {
15  if(light==red)
16   method1.3();//the method mentioned in 1.3
17  else
18   if(Q_block.empty()){
19    if(time_remain>T_safe)
20    //same as the code block 1
21    else
22     if(!pass_check())
23     //same as the red light case
24    }
25   else
26    if(time_remain>T_safe)
27    acc_to_speed(v_dis);
28    else
29     if(!pass_check())
30     //same as the red light case
31    }
32
33 }
```

The plantform will traverse the queue Q_{in}, and during each iteration, the acceleration of the vehicle in processing will be set according to the model given above.

1.6 Connected-vehicle control strategy without the inter-vehicle communication model

2 YOU KNOW WHAT

2.1 the name of 1.6 should be changed

2.2 give a implementation of car-following method