**Road Traffic Simulation for Analysing Travel Time for Synchronized and Unsynchronized Traffic Signals**

**Final report**

**Team No. 24**

**20 February 2015**

Chandra Khatri

Hidekazu Kose

Hao Luo

**1. Project Description**

The purpose of this project is to investigate a traffic signal pattern which could reduce travel time of vehicles on Peachtree Street within a given zone between 10th Street and 14th Street (Fig.1), obtain the distribution of travel time for all vehicles transversing the designated zone within a given time period and compare that for two scenarios. The first case assumes that all the traffic signals in the intersections are synchronized, meaning the phase among signals are fixed. The other case is an unsynchronized one, where the signals operates with a randomly assigned phase. Essentially, the former scenario is one special case of latter: the fixed phase can also be one of the random phases. Thus the synchronized phase can be chosen from unsynchronized pool. The one which outruns the rest in several unsynchronized simulations can be the choice for the synchronized simulation.

Furthermore, we simulate the overcrowded situation that is often observed on actual Peachtree street at a peak time. To accurately reproduce the situation in the simulation, we model spillover where drivers can not across an intersection because the road in front of the vehicle is occupied by preceding vehicles.

The static road network is obtained through third party information (Google Map). There are in total 5 intersections in the area of interest, of which 4 are signalized and 1 is unsignalized( intersection of 13th Street with Peachtree Street). The sequence for each signal follows green-yellow-red cycle, with the duration for the timing pattern of each signal fixed For both scenarios. The specific timing pattern for all the signals can be found in the Excel file signalTiming.xls. Along this part of Peachtree Street, there are 6 sections, with each section have different length and number of lanes. A general process for an individual vehicle is depicted as follows:

1) Enter the zone of interest: The vehicle can enter from originating zones shown in Fig.1 ;

2) Go through intersection or turn right or turn left if signal turns green;

3) Decide whether stop or continue its movement if signal turns yellow;

4) Stop if signal turns red and continue its movement as in 2 if signal turns green again;

5) Leave the zone of interest: as in 1, the vehicle can leave into four directions.

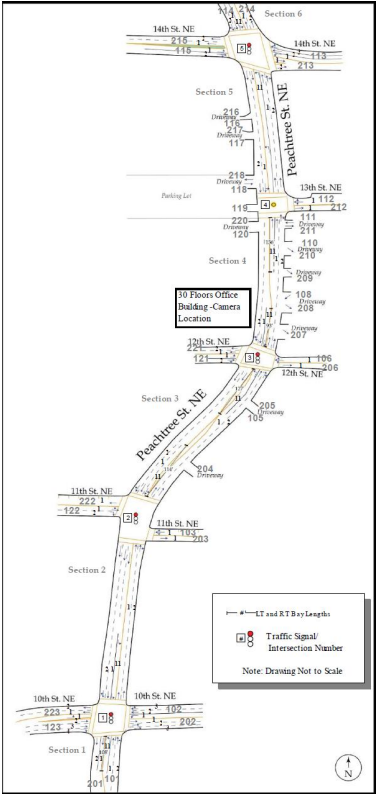


Fig. 1. Real map illustration

**2. Conceptual Model**

We discuss our conceptual model in this chapter. First, we mention model assumptions and simplifications. Then we show input and output of our model. Details of them are further discussed in Chapter 4 and Chapter 5 respectively. Then we show components and interactions between the components of the simulation model.

**2.1 Assumptions and simplifications**

Assumptions and their corresponding simplifications are shown with (A) and (S) prefix.

**2.1.1 Simplification of behavior of vehicle**

(A) Drivers don’t push an accelerator

-> (S) The velocity of each vehicle is drawn randomly when they arrive at the zone of interest and it does not change after that .

-> (S) Vehicle reaches a terminal velocity immediately after it leaves from an intersection.

-> (S) Vehicle immediately stops in front of an intersection if traffic signal is red.

(A) Driver ignores yellow signal.

-> (S)Each phase of a yellow signal is regarded as a part of green signal phase.

(A) Drivers do not make lane change.

-> (S)Vehicle does not pass the car in front.

-> (S)Vehicle keeps the same lane position even if the neighboring lane is empty.

(A) Drivers do not change their velocity or behavior by other drivers. (i,e, drivers are mutually independent.)

**2.1.2 Simplification of input**

(A) All random event such as arrival of vehicle, probability of turning event at an intersection are stationary stochastic process.

(A) Phase delay from an adjoining traffic signal(we call it “offset” and express it with ta .) is constant and the value of the offset is same for all signals. For example, if the first signal turns green at t = 0, the second and the third signal turns green at t = ta and t = 2ta respectively.

(A) There is no vehicle in the zone of interest when the simulation starts.

**2.1.3 Simplification of output**

(A) Transient period of the simulation is small enough to be ignored.

**2.1.4 Simplification of parameters**

(A) Approximation error of following parameters are small enough to be ignored.

->(S) All traffic signal has the same cycle time(100 seconds) and has the same duration of green and red interval (60 seconds and 40 seconds respectively).

->(S) Travel time to pass through an intersection, first and sixth segments are short enough to be ignored.

**2.2 Input and output of the model**

**2.2.1 Input**

The offset of signal is one of the most important input because the objective is of this project is to investigate the influence of the offset on average travel time of vehicle. The offset of signal is constant at every trial in synchronized pattern. On the other hand, the offset is randomly generated in synchronized pattern at each trial of unsynchronized pattern. The range of the offset is between 4 sec and 15 sec.

The following random variables are also inputs of the simulation.

1. Arrival rate of vehicle
2. Velocity of vehicle
3. Length of vehicle
4. Probability of right or left turn at an intersection

We model these variables such that average traveling time of vehicle in the simulation is close to NGSIM[1] data as shown in Chapter 4.

**2.2.2 Output**

In our simulation, three cases are considered: synchronized case, unsynchronized case and unsynchronized spillover case. The final output will be distributions of travel time, velocity etc. for all of the these cases, as well as comparisons among them and with NGSIM data.

**2.3 Simulation Process**

In this project, we will build simulation program based on the Process-oriented worldview[2][3]. Therefore we need to identify the behavior of individual vehicles and define components which interact with vehicle during simulation process. Graphical modeling language such as SysML(Systems Modeling Language) allows a better understanding of our simulation model[4]. Hence we draw block definition diagram and sequence diagram in SysML to represent the behavior and components of the model.

**2.3.1 Components of the model**

Fig. 2.1 shows block definition diagram of a components of the simulation program (Note: Each block in Fig.2.1 is not necessarily implemented as a class in source code). Each rectangle(block) represents a component of the simulation model. Value properties and operations of components are shown in compartments of the block. Arrows with filled diamond head express composition a whole-part relationship and the multiplicity of the part end is illustrated near the arrow heads. As shown, our simulation model is composed of a number of vehicles, five roads, four intersections and one event scheduler.

**Vehicle block**

Vehicle block represents a vehicle which pass through the zone of interest. The block has value properties such as vehicleID, entryPoint, exitPoint, velocity, length and arrivalTime. Each vehicle is distinguished by unique vehicleID. A location where the vehicles enter and exit the zone of interest is expressed as entryPoint and exitPoint, and passway of the vehicle can be obtained from them. Time intervals during which the vehicles arrive at and leave from each intersection are stored in timeStamp value property.

**Road block**

Road block represents roads between intersections. As noted in model assumptions and simplifications part, velocity of a vehicle is constant throughout its travel, however it is different for different vehicles and is randomly drawn. This means that traveling time of a vehicle on the roads/lane from one intersection to next intersection can be derived as a division of segmentLength by vehicle velocity (it is the time on road between two intersections hence does not include the waiting time at signal and queue). The method named getTravelTime calculate the total traveling time for a vehicle given its arrival time, direction, start and exit location and the global state variables (signalStates, Queues)

**Intersection block**

Each intersection has Northbound and Southbound TrafficLane and TrafficLight blocks.

**TrafficLane block**

TrafficLane block has listOfWaitingVehicle property to which vehicles waiting in the traffic lane are added. While a vehicle pass through an intersection, stateOfTrafficLane value property comes into occupied state. After the vehicle leaves the intersection, stateOfTrafficLane gets to empty state. The operation named getTimeToPass returns time interval that the vehicle pass through the intersection with considering the preceding vehicles.

**TrafficLight block**

TrafficLight block has signal\_color value property which represents color of traffic light (it is a boolean and 1 represents green/yellow while 0 represents red). Time interval of green and red light is defined by cycle\_timing\_list value property. The operation named getTimeToGreen returns time interval during which the vehicle of interest has to wait for green signal.

**Global State Variable**

Global state variables defines global variables that are shared with some classes.

**Event handler**

Event handler manages events of vehicle instances.

**2.3.2 Interactions between components**

Fig. 2.2 shows a sequence diagram which illustrates a part of the simulation process. In Fig. 2.2, instances of the blocks defined in Fig. 2.2 are illustrated as rectangles and interactions among them are represented by solid or dashed arrows. Time flows from top to bottom of the diagram.

First, the event handler creates an instance of the vehicle or retrieve existing instance from future event list(createVehicleInstance/getVehicleFromFEL). Second, the vehicle instance calls getTimeToGreen and getTimeToPass operation to obtain time intervals during which the vehicle waits for a green signal and all preceding vehicles pass through the intersection. The vehicle instance also calls getTravelTime operation to get time to travel one intersection to the next one. These operations are repeatedly called until the vehicle leaves the zone of interest. After the vehicle leaves the zone, it calls getTravelTime operation to record the obtained time intervals.

Event Handler pops the entities/vehicles from the future event list and decides, when next vehicle enters the zone of interest and calls advanceTime operation to advance the simulation time by the time obtained in the previous operation. Then it creates another instance of vehicle(vehicle2) and continues the same processes until the simulation time reaches a finishing time.

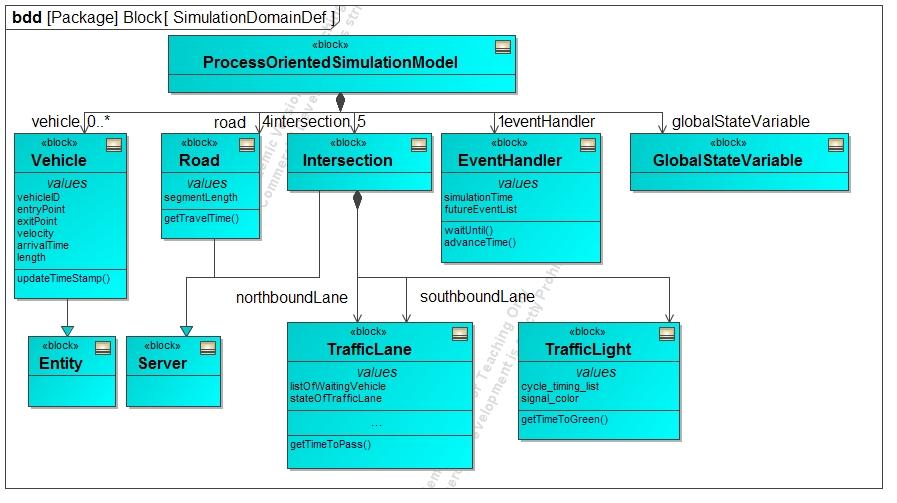


Fig. 2.1 Block Definition Diagram of the simulation model.

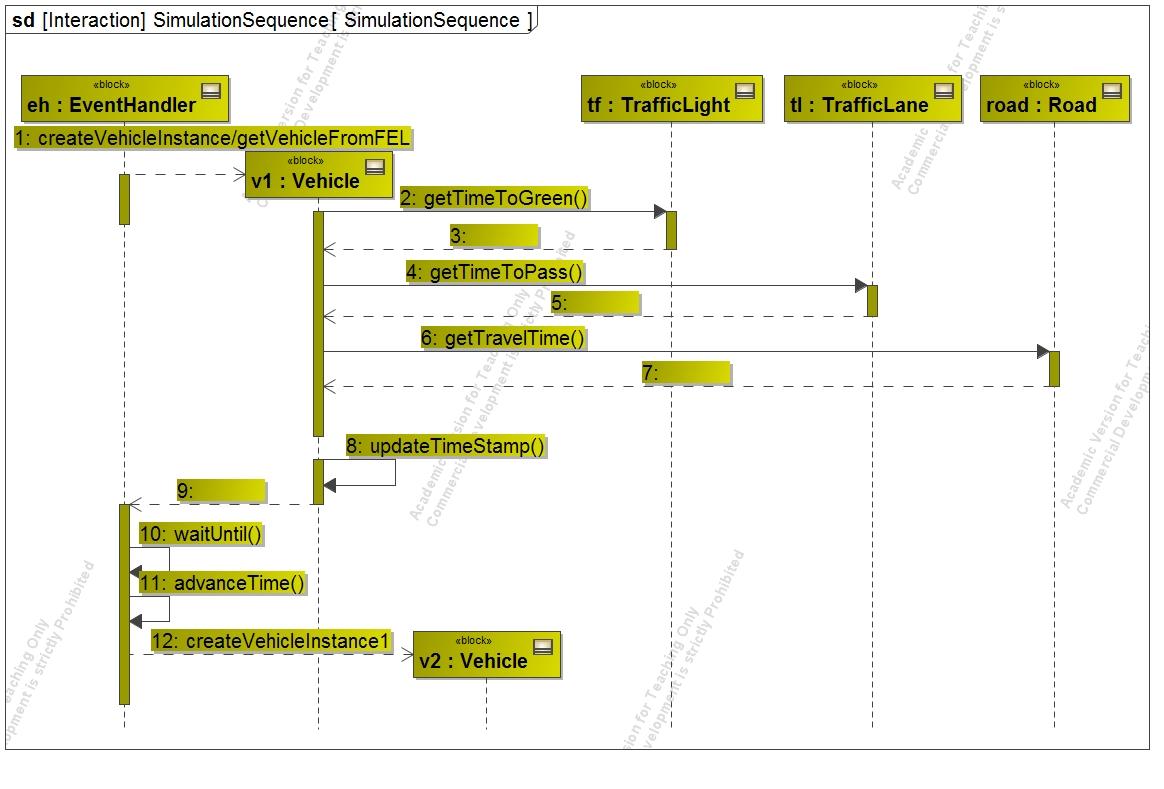


Fig.2.2 Sequence Diagram of the simulation model.

**3 Simulation Code**

**3.1 Tools**

Python is being used for modelling the entire Process-oriented simulation model. Velocities and all other variables are derived from the NGSIM’s data, which are also mentioned in the input analysis part of the report. Definitely Priority Queues are used to perform entire Simulation. Although, we have created our own code for Priority Queue and it is working but we have used Python's inbuilt Priority Queue implementation for performance boost as we run 500 simulations with spillover.

**3.2 Code Overview**

Following are the important methods that are used in the simulation:

1) FELqueueing : FEL generator for each previous FEL and state variables and other dependencies

based on Simulation Engine/Even handler.

2) SimulatingOneEntireTrip : Basically serving each FEL/Queue and updating the state variables.

3) Simulation\_Engine: Main Event handler.

4) MaintainTurns : Maintain turnings with given rates from NGSIM data.

5) TrafficSim : Initial FEL handler/generator.

6) ProcessBasedSimulation : Main method, basically entry to the code.

**3.3 Code Explanation:**

“ProcessBasedSimulation” calls “trafficSim” method which actually generates FEL with arrival events of each vehicle with a time-stamp for the entire simulation period (including vehicles from sides). Once all the vehicles are gathered to be served with their individual time stamp and FEL, “Simulation\_Engine” method is called which starts the process oriented simulation and serves vehicle by vehicle for each FEL. It then handles/updates the state variables using “simulationOneEntireTrip” (basically on FEL as we have multiple FEL, which also get updated during the course of simulation). The simulatingOneEntireTrip uses FELqueueing method which is the last method and performs each entity.

**3.4 Input Parameters**

Following are the input parameters and vehicle properties are considered in current project:

segment\_length\_list = [442, 412, 354, 344]

returnvehicle\_arrival\_rate = [0.60 ,0.08, 0.04, 0.04, 0.04, 0.06, 0.70]

turning\_rate = [0.5,0.02,0.05,0.05,0.5, 0.45,0.2,0.05,0.15,0.5]

cycle\_timing\_list = [40, 60]

signal\_offset = random.randint(4,15)

vehicle\_mu = 23

vehicle\_sigma = 3

vehicle\_length\_mu = 16

vehicle\_length\_sigma = 1

offset\_between\_vehicles = 10

The units used for the current simulation project are feet for distance and length, feet per second for vehicles, seconds for time. As mentioned above, these values are obtained and later on validated and verified from NGSIM data.

**4. Input analysis**

In this chapter, we discuss how we model the following inputs.

1. Arrival rate of vehicle
2. Velocity of vehicle
3. Length of vehicle
4. Probability of right or left turn at an intersection

**4.1 Arrival rate of vehicle**

We first investigate how many vehicles arrive at or depart from each intersection. Fig.4.1 represents number of vehicles that enter the zone of interest from intersections (top) and leave from the zone (bottom). As shown above, more than 83% of vehicles came from the first intersection(Originating zone iD is 101,102,123) or fifth intersection(Originating zone ID is 113, 114, 115). Further, more than 86% of vehicles depart from first(Destination zone ID is 201, 202, 223) or fifth intersection(Destination zone ID is 213, 214, 215). From this result, we focus on the probability of the arrival event at the first and fifth intersections. The inter-arrival time of vehicles at the first intersection is shown in Fig.4.2. we assume that the distribution of time intervals of the arrival events follows an exponential distribution[5][6] and estimated the rate parameter at 0.6 from the NGSIM data. As shown in Fig.4.2, the number of vehicles from second, third fourth intersections are not large enough to estimate the distribution. Thus, we assume that probability of the arrival of vehicle is uniformly distributed and the probability of occurrence is selected so that the total number of vehicles that arrived at the zone of interest is close the NGSIM data. Further discussion about the rate of arrival event is shown in Appendix B.1.

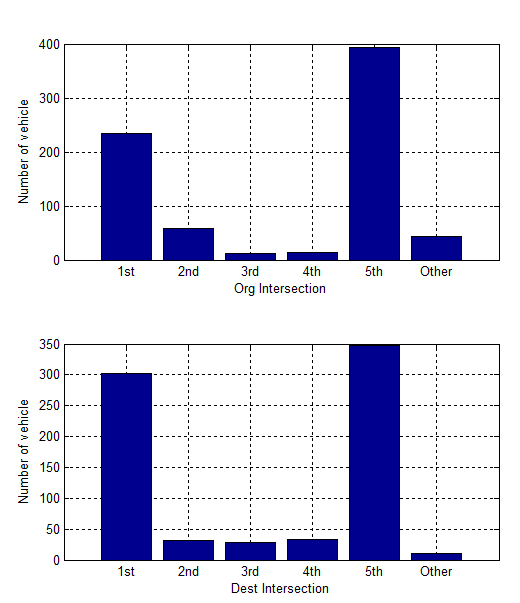


Fig.4.1. Number of vehicle at each intersection (top:originating zone, bottom:destination zone)

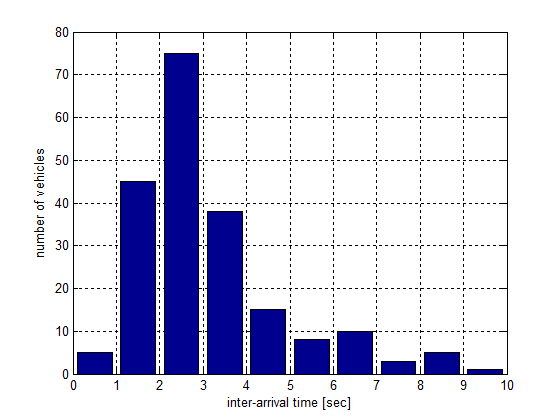


Fig.4.2. Inter arrival time of vehicle at originating zone 114

**4.2 Velocity of vehicle**

The average travel time under a certain traffic signal pattern would depend on the velocity of the vehicle. In addition, distribution of the velocity would influences the congestion condition of the road and it would affect the travel time. Therefore, velocity of vehicle should be modeled as a random variable to precisely predict the traveling time. Since the length of an intersection is shorter than that of road between intersections(i.e. section), we only consider the velocity when a vehicle is driving through a section. Further, as NGSIM data includes velocity of vehicles when they stop waiting green signal, we exclude data if velocity is less than 1 ft/sec. A histogram in Fig.4.3 illustrates a distribution of velocity. We assume that distribution follows a normal distribution and select the standard deviation so that the average traveling time of vehicles in simulation to be close to those in NGSIM data. Further discussion about velocity of vehicle is shown in Appendix B.2.

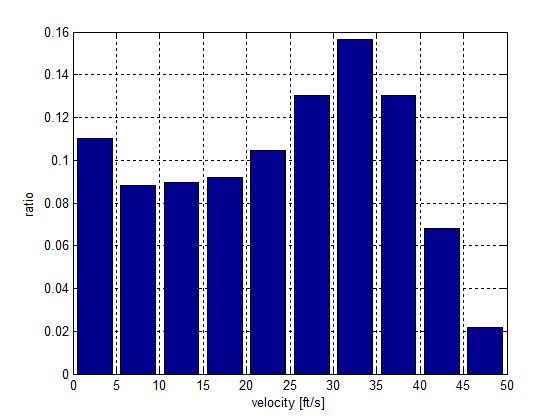


Fig.4.3 Velocity of vehicle

**4.3 Length of vehicle**

We calculate average travel time considering the length of a line of vehicles waiting at an intersection. Further, we model spillover. Therefore, the length of the vehicle should be an input parameter of the simulation. Since average length of vehicle depends their types, we first count the number of vehicles per their type. According to the given NGSIM data, vehicles are classified into motorcycles, automobiles, tracks. Since number of automobiles are larger than others (number of automobile is 738, motorcycle is 1, track is 14), we only consider the length of automobiles. Fig.4.4 shows the histogram of the length of automobiles. The mean and standard deviation of the length of the vehicles are 16.05 feet and 2.87 feet. We assume that the distribution of the length of the vehicles follows a normal distribution.

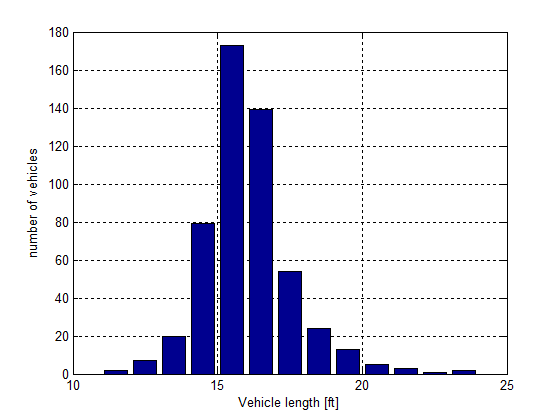


Fig.4.4 Length of vehicles

**4.4 Probability of right or left turn at an intersection**

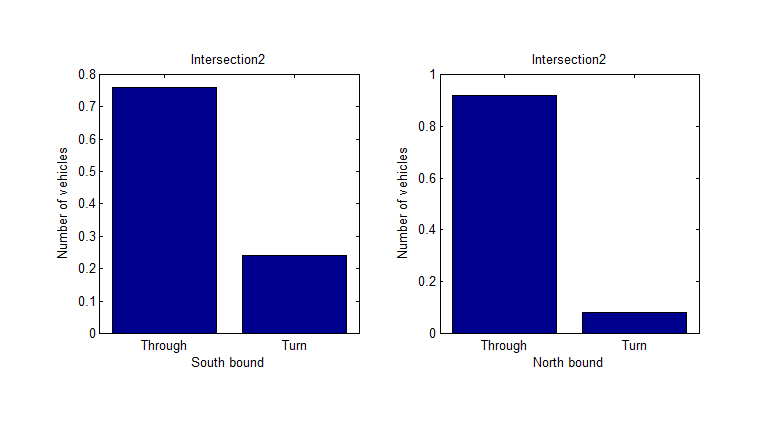
As noted in section 4.1, more than 83 percent of vehicles leave from northernmost and southernmost intersections. Thus, the travel time of the vehicles that leave intermediate intersections may have less influence on the average traveling time over all vehicles. However, if destinating zones are constrained in northernmost and southernmost intersections, the road will be more crowded than it actually is and the difference of congesting condition of the road influences the average traveling time of other vehicles because probability of spillover will increase and a line of waiting vehicle at an intersection should be longer. Thus, we model a departure of vehicles at the intermediate sections and obtain the probability of the departure event at each intersection from NGSIM data by distinguishing the direction of vehicles. Fig. 4.5 shows the number of vehicles that go through, turn left and turn right at the second intersection.

Fig.4.5 Number of vehicles that go through and turn at the second intersection(left:southbound vehicles, right:northbound vehicles).

**5 Output analysis**

**5.1 Model verification:**

Offsets in the range of [4,15] are assigned randomly to each simulation. A test for the random generator in python was conducted in Matlab. It turns out the test is accepted with 95% confidence. However, the p-value is 0.1603, which casts doubt on the validity of the random generation. One explanation for this is our sample size is not big enough. In total, 500 simulations were ran. In our modeling, the simulation that has the lowest mean travel time was chosen as the synchronized case. The rest of 500 simulations were categorized as unsynchronized case. Except for these two scenarios, we took a step further to model an extreme case: unsynchronized case as will be discussed below.

In this modeling, velocity is also assigned to vehicles randomly based on normal distribution with mean value and standard deviation obtained from NGSIM data. To verify the effectiveness of this random generator, a statistic test was performed in Matlab for synchronized case, with acceptance of 95% confidence and p-value 0.3075. It is worthwhile to note here that NGSIM velocity distribution doesn’t follow normal distribution, as shown with red line in Fig.5.1.1. The reason why we don’t use NGSIM distribution can be found in Appendix B.2. Nevertheless, as verification purpose, we also perform a test about the output velocity distribution, as indicated as

bar chart in Fig.5.1.1. The result seems to be reasonably good.

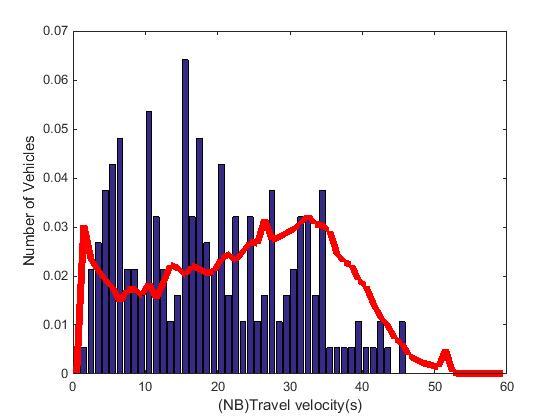


Fig. 5.1.1 Travel velocity distribution from NGSIM data and fitted simulation data. Blue bar denotes simulation data while red line for NGSIM data.This comparison is for verification purpose.

Furthermore, within the observation time from VGsim.data during evening period, the total number of vehicles going from North to South or South to North is consistent with our output total number of vehicles as this given amount of time (around 450). And the turning rate at each intersection for final outcome agrees well with the observed turning rate.

**5.2 Model validation**

Aforementioned, three scenarios are considered here: synchronized signals, unsynchronized signals and unsynchronized spillover. The synchronized case has an offset of 4 seconds. For the sake of explanation, the simulation of worst performance was chosen as the unsynchronized case, whose offset is 15 seconds.

Comparisons of synchronized case with NGSIM data are shown in Fig. 5.2.1 , Fig. 5.2.2 and Fig. 5.2.3 . Fig. 5.2.1 shows the travel time distribution for NB and SB, while Fig.5.2.2and Fig.5.2.3 reveals that distribution of NB and SB respectively. A good fit of our simulation results with NGSIM data was observed, though with some discrepancy, which may be attributed to our assumptions and simplifications.

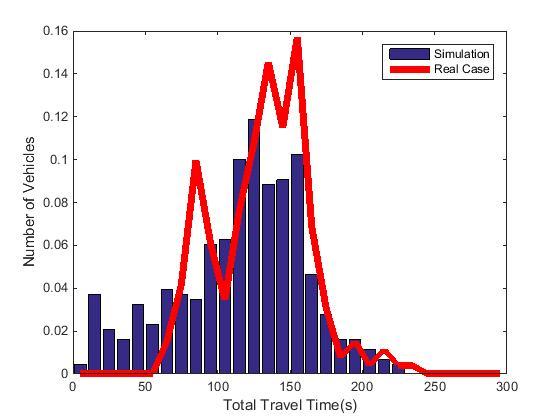


Fig.5.2.1 Travel time distribution of vehicles going southbound and northbound for synchronized case.

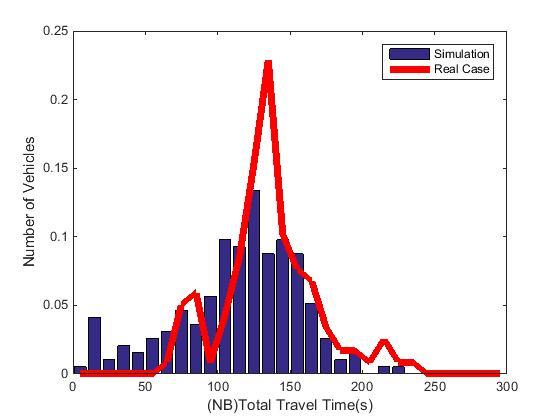


Fig. 5.2.2 Travel time distribution of vehicles going northbound for synchronized case.

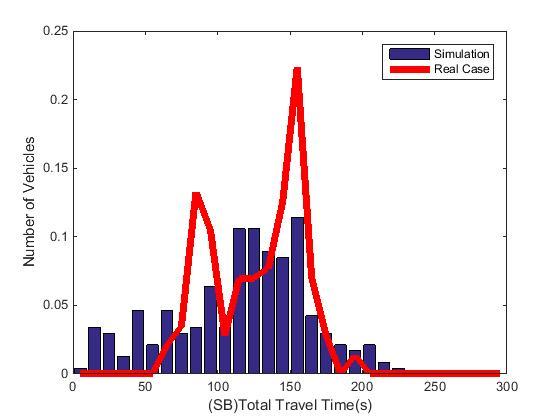


Fig.5.2.3 Travel time distribution of vehicles going southbound for synchronized case.

Similarly, comparisons of unsynchronized case were shown in Fig 5.2.4, Fig 5.2.5 and Fig 5.2.6. Obvious deviation of both mean and peak value from NGSIM data was observed.The different behaviors of synchronized and unsynchronized case are consistent with our experience. Basically, the real signal pattern and phase should be able defeat other candidates so that it can stand out and serve the best for traffic. In this sense, we can view NGSIM data as synchronized case. This is why the performance of simulation employing synchronized signal phase excels that using unsynchronized signal phase. This difference thus validate a model to a certain degree.

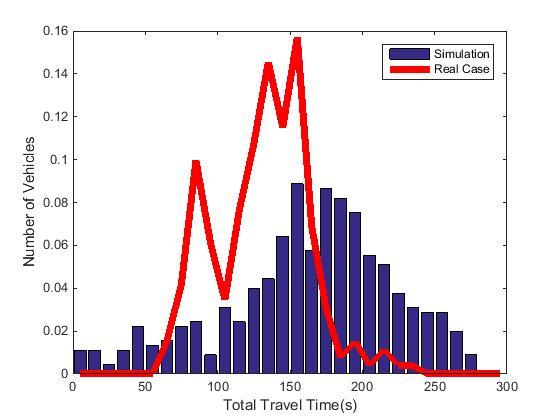


Fig. 5.2.4 Travel time distribution of vehicles going southbound and northbound for unsynchronized case.

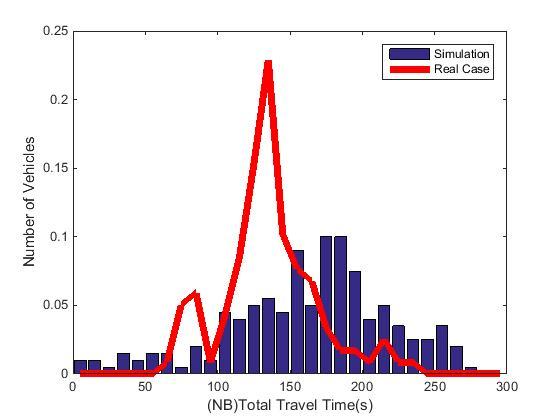


Fig. 5.2.5 Travel time distribution of vehicles going northbound for unsynchronized case.

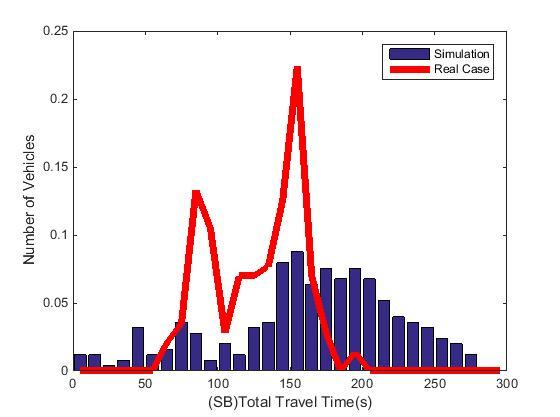


Fig.5.2.6 Travel time distribution of vehicles going southbound for unsynchronized case.

Apart from the above synchronized and unsynchronized cases, a unsynchronized spillover experiment was also conducted by limiting the queue size, so that queue could be filled up fast. Fig.5.2.7 shows the result. As we can see from this figure, it took most of the vehicles several hundred seconds to finish the travel from south to north or vice versa. What is worse, chances are that even travel time of 1800 seconds is possible. It can be imagined that at very high traffic situation, most of the drivers have to wait on the road anxiously. The extreme case was possible in that some lane was so fully occupied but vehicles were keep coming and few turnings at the intersections.

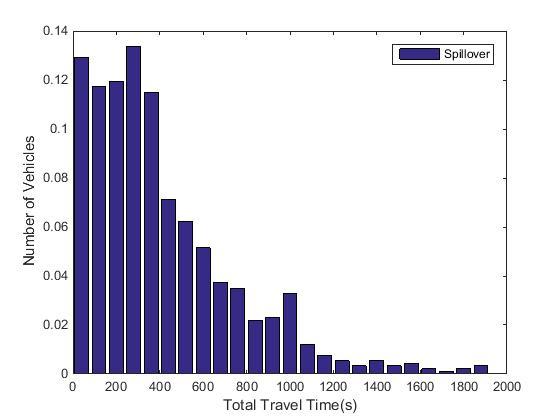


Fig. 5.2.7 Travel time distribution of vehicles going southbound and northbound for unsynchronized spillover case.

**5. 3 Observation**

As already seen in the Verification part, the outcome from synchronized and unsynchronized case is very different. A direct comparison between these two cases are shown in Fig. 5.3.1. Further in this part, we performed t-student test to see the significance of that difference. It should be noted that the unsynchronized case is the same as we use for verification purpose. The interval with 95% confidence is 47.8~48.3 with acceptance. This degree of difference can also be approximately seen from the corresponding travel time of peak values in Fig. 5.3.1.

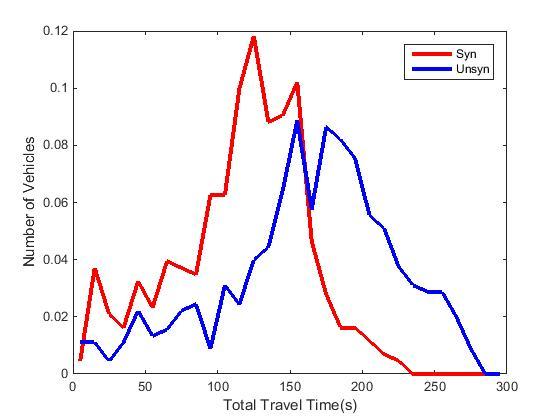


Fig.5.3.1 Comparison of travel time distribution between synchronized and unsynchronized case.

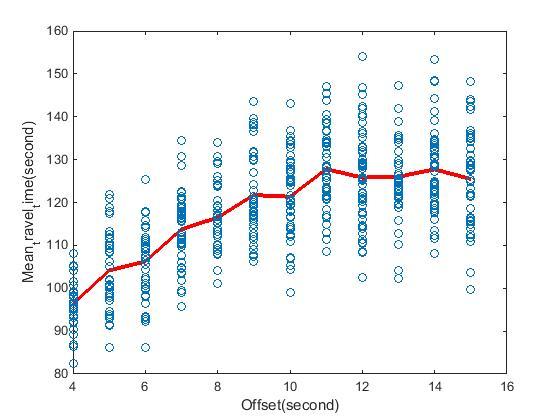


Fig. 5.3.2 A general trend of mean travel time versa offset

Further, in Fig.5.3.2, a general increasing trend of mean travel time as offset becomes large. The dots are the scattered simulation results for each of offsets while the red line connects the average performance for each offset. A general prediction that can be made from this observation is that an even smaller offset may have a better performance than the current synchronized case. And a plateau shows after offset becomes 12 seconds, which denotes the worst case. Since the pattern for each signal is set, we can conclude that a cyclic response of mean travel time versa offset can be seen if the pool of offsets is large.

**6 Conclusion**

In summary, based on the NGSIM data, we generated the input for the simulation, like the vehicle arrival rate, vehicle length, vehicle turning rate, vehicle velocity, signal patterning, etc. Verifications and validations for our model are carried out. Specifically, over 500 simulations are performed, among which synchronization , unsynchronized and unsynchronized spillover cases are considered. The synchronized case seems to fit well with the NGSIM data while there is a significant deviation of unsynchronized simulation from NGSIM data. Statistically, the travel time of the synchronized case excel unsynchronized case with around 50 seconds at confidence level 95%. An increasing trend of mean travel time as offset goes up indicates that a small offset would have a beneficial effect on facilitating the traffic.

**7** **References**

1. Feng, Y., Hourdos, J., Davis, G., Collins, M., “Vehicle Probe Based Real-Time Traffic Monitoring on Urban Roadway Networks”. Minnesota Traffic Observatory, Department of Civil Engineering University of Minnesota
2. [Schwetman, H.D](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Schwetman,%20H.D..QT.&searchWithin=p_Author_Ids:37299532800&newsearch=true)., “Introduction to process-oriented simulation and CSIM”. [Simulation Conference, 1990. Proceedings., Winter](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=185), 9-12 Dec 1990, 154 - 157.
3. Healy[,](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Schwetman,%20H.D..QT.&searchWithin=p_Author_Ids:37299532800&newsearch=true) K.,Kilgore, R., “Silk: a Java-based process simulation language”.Proceeding WSC '97 Proceedings of the 29th conference on Winter simulation Pages 475-482.
4. S. Friedenthal , A. Moore and R. Steiner, “A Practical Guide to SysML, Second Edition: The Systems Modeling Language.” , Morgan Kaufmann/OMG, Elsevier
5. Chandrasekaran, B., “Survey of Network Traffic Models”
6. Camps, F., Thibault, E., Harasse, S., “Statistical Distribution of Traffic Sources in Network Simulation Tools”. Proceedings of the World Congress on Engineering and Computer Science 2008 WCECS 2008, October 22 - 24, 2008, San Francisco, USA
7. BP Zeigler, H Praehofer, TG Kim., “Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems”. Academic Press, Inc, 2000.

|  |
| --- |
|  |

### 

|  |
| --- |
|  |
|  |

**Appendix A. Event oriented simulation**

**A.1. Overview**

We also build an event oriented simulation[7] program for our better understanding of course materials in addition to the process oriented program shown in the above chapters,. This program is built based on the same model assumptions as the process oriented program. For this purpose, we also build a conceptual model of the simulator by using graphical modeling language (System Modeling Language, SysML). Block definition diagram(BDD) defines components of the program and sequence diagram defines interactions between the components. In our event oriented simulation program, components shown in BDD are implemented as a class.In this chapter, we show how our event oriented simulation program works by using BDD and sequence diagram of SysML.

**A.2. Block definition diagram**

Fig.A1 shows BDD of the event oriented simulation program. Each rectangle(block) represents a component of the simulation model. Value properties and operations of components are shown in compartments of the block. Arrows with white triangle head express a generalization relationship and specialized component is illustrated at the arrowheads. Arrows with white diamond head express an association relationship and referenced components are shown at the arrowheads. Though we also build a class which record behavior of vehicles during simulation and draw figure based on them, we do not put them to make the diagram concise.

**A.2.1 Scheduler class**

Scheduler class manages a future event list. The future event list is a linear list and sorted according to the time when the event should be executed. Other classes such as event class or Simulation class can add new event by calling addEvent operation. If there are events whose scheduled time are same as the new event, Scheduler class compares priorities of events and insert the new event before the existing events in the future event list if the new event has higher priority. After an event is executed, the event is excluded from the future event list.

**A.2.2 Simulation class**

Simulation class calls events and manages simulation time. First, this class retrieve an event which is listed the top of the event list. Then, the class advance simulation time by the time when the event should be executed and execute the event. This process is repeated until the simulation time reaches an end time.

**A.2.3 Intersection class**

Intersection class has queue, signal color, state of occupancy of an intersection as a state variable. Behavior of vehicles depends on the state variables and Event class shown later calls operations to change the state variables. For example, when vehicles arrives an intersection, they are listed in the queue and the vehicles can be dequeued only when the signal color is green and the intersection is empty.

**A.2.4 Event class**

The simulation program has four event class that inherit Event class. SignalEvent is an event where traffic light changes from red to green or green to red. ArrivalEvent enqueues vehicles and DeparetureEvent dequeue vehicles and change the state of occupancy to occupied state considering with the traffic light. In ServiceCompleteEvent set the state of occupancy as empty state when vehicle leaves the intersection.

**A,2.5 Event owner class**

In the simulation program, we consider behavior of multiple entities such as vehicles and traffic signals. To manage their behavior and state variables, all events are associated with EventOwner who has unique ownerID.

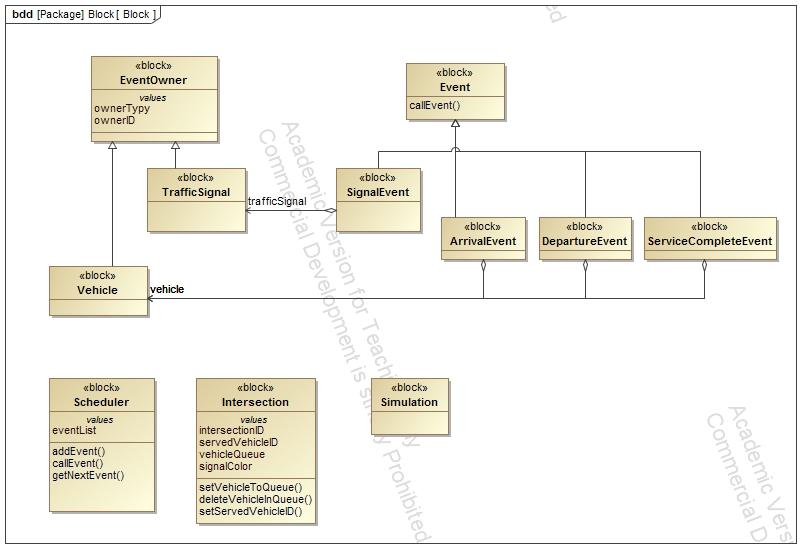


Fig. A1 Block definition diagram of the simulation program

**A.3. Sequence Diagram**

We introduce a simple scenario to show how our event oriented program works. In this scenario , we assume that there are only two vehicles going through the same direction and there is only one intersection. When the first and second vehicles arrive at the intersection, the traffic signal is red. Therefore, they have to wait until the signal turns green. When signal turns green, the first vehicles starts to pass through the intersection and after the first vehicle leave the intersection, the second vehicle starts. A sequence diagram which illustrates the scenario is shown in Fig. A2. In Fig. A2, instances of the blocks defined in Fig. A1 are illustrated as rectangles and interactions among them are represented by solid arrows. Time flows from top to bottom of the diagram. To make diagram concise, reply messages from receivers are omitted.

When the simulation starts, Simulation class add arrival event of the first vehicle(1) and signal event(2) to the future event list. Then, it get next event(3) (in this case, returned event is the arrival event of the first vehicle) and advance time by the time when the event should be executed(4). Simulation class also calls ArrivalEvent(5). ArrivalEvent class add it’s owner (i.e. vehicle) to queue at an intersection(6). Then the ArrivalEvent class add arrival event of the second vehilce(7) and DepartureEvent of the first vehicle(8). When Simulation class calls DepartureEvent of the first vehicle(9), the first vehicle does not cross the intersection because the traffic light is red. Thus, the DepartureEvent schedule the next DepartureEvent for the first vehicle(10). Simulation class calls ArrivalEvent of the second vehicle(11) that is scheduled at (7) and the ArrivalEvent add the second vehicle to the queue. After the second vehicle arrives, Simulation class calls SignalEvent(13) that is add to the future event list at (2). SignalEvent changes signal color from red to green(14) and schedule next SignalEvent where the traffic light turn green to red(15). The DepartureEvent of the first vehicle is called when the traffic light is green(16) and the event calls operation of the Intersection class to dequeue the first vehicle(17) and change the state of occupancy to occupied state. Further, the event schedule ServiceCompleteEvent for the first vehicle(19) and DepartureEvent for the second vehicle(20). After the state of occupancy becomes empty state by ServiceCompleteEvent(21), the second vehicle leaves from the intersection(22).

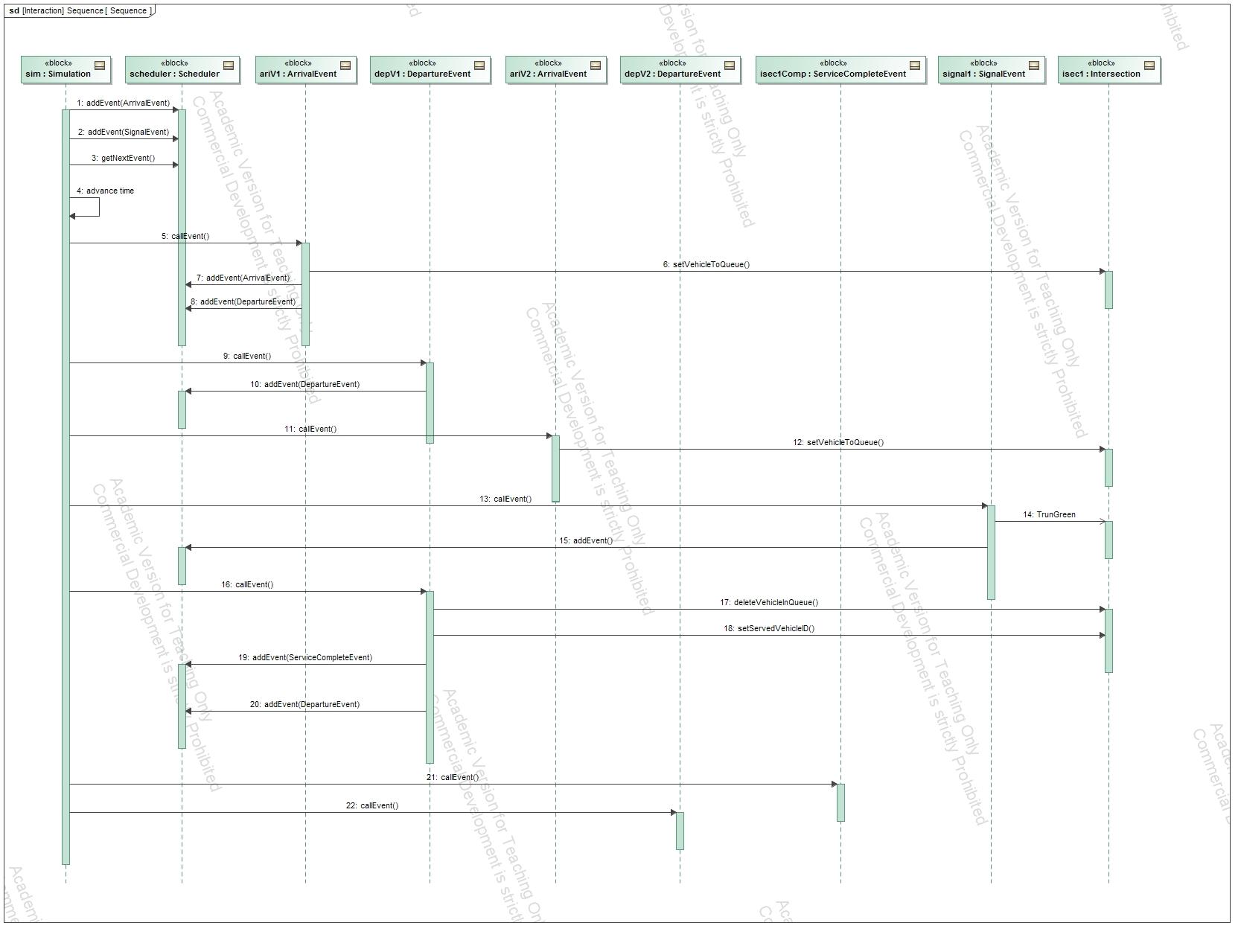


Fig. A2 Sequence diagram of simulation process

Fig. A3 represents time history signal color and queue at an intersection. In Fig. A3, horizontal line represents simulation time. Green and red circles illustrates color of traffic light at the time. Numbers in black shows IDs of vehicles listed in the queue and numbers in red represents an ID of a vehicle who is passing through the intersection. In this simulation, vehicle arrives at every four seconds and it takes four seconds for vehicle to pass through the intersection. The first vehicle(ID is one) and the second vehicle arrive at 4 sec and 8 sec respectively and they wait until 9 sec when the traffic light turns green. The first vehicle occupies the intersection from 9 sec to 12 sec and then leaves from the intersection. The second vehicle goes to the intersection at 13 sec and leave at 16 sec.

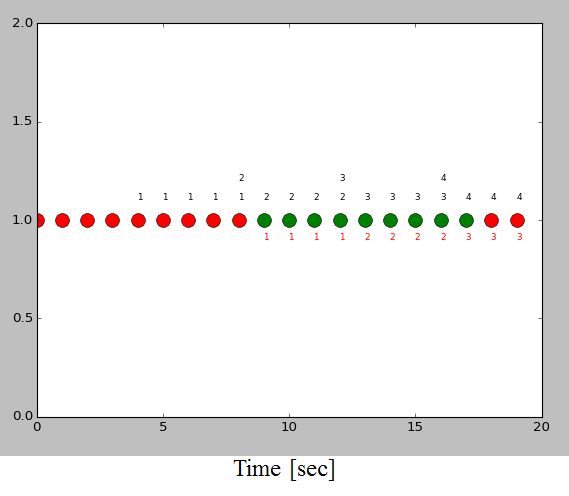


Fig.A3. Time history of signal color and queue at an intersection

**Appendix B.**

**B.1 Arrival rate of vehicle**

As mentioned in section 2.1.2 and 4.1, we assume that the arrival event of vehicle is a stationary stochastic process and the inter-arrival time of vehicle follows the exponential distribution. However, the assumption is not valid in some intersections.

Fig.B1 shows a time history of arrival events of vehicles at originating zone 114 obtained from NGSIM data.The horizontal axis represents the observation time and vertical axis shows the number of vehicles that arrived at the observation time. As shown, arriving rate changes periodically at every one hundred seconds. This means that the arrival event in the originating zone 114 is not a stationary stochastic process. We guess that this periodical change of arrival rate is caused by northern traffic signals of originating zone 114. Thus, by considering the change of the distribution, we could make a simulation model much closer to the observation.

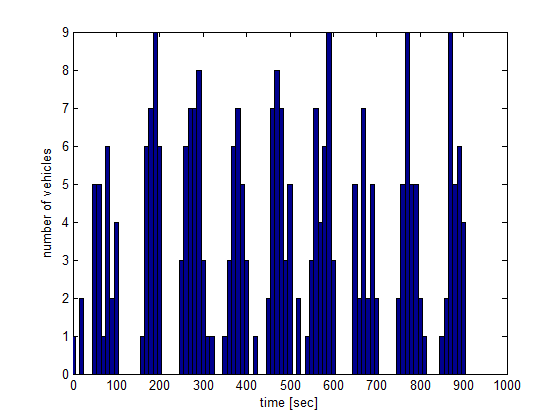


Fig. B1. Time history of arrival event in NGSIM data.

Fig.B2 illustrates a histogram (blue bar) of inter-arrival time at originating zone 114 from NGSIM data. As shown in the figure, the distribution has the peak at not zero but between two to three seconds. Therefore, NGSIM data does not give us enough evidence that the distribution of inter-arrival time follow the exponential distribution. To examine other alternatives, we conduct a Chi-Square Good Fitness Test under the assumption that the distribution of the inter-arrival time follow the gamma distribution. In Fig.B2, blue line illustrates a probability distribution function of gamma distribution with maximum likelihood estimation from NGSIM data (shape parameter k is 5.517, scale parameter theta is 0.4892). The result of Chi-Square Good Fitness Test shows that the null hypothesis is not rejected with 95% confidence. Therefore, in so far as originating zone 114, it would be better to use the gamma distribution as probability of the arrival event.

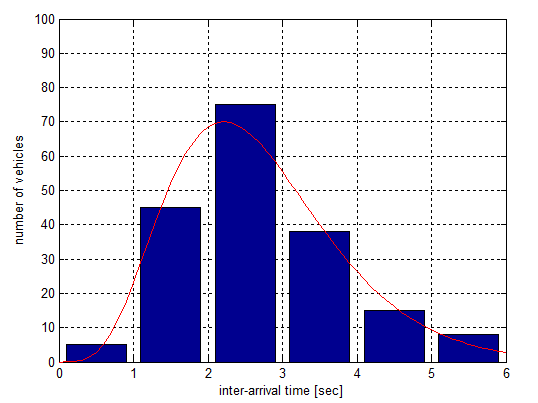


Fig. B2. Inter-arrival time of vehicles.

**B.2 Velocity of vehicle**

As mentioned in section 4.2, we assume that distribution of the velocity follows the normal distribution and we select the parameter so that an average travel time of the simulation to be close to that of NGSIM data. However, velocity of vehicle in NGSIM data seems uniformly distributed between and as shown in Fig.B3. We build random variable generator which yields velocity of vehicle by using inverse function of cumulative distribution function. Fig.B4 shows a histogram of cumulative distribution function and red line represents approximating curve. The approximating curve is fifth order polynomial function and obtained by a least square algorithm. We also build the random variable generator for other intersections in a similar manner and conduct simulation. However, we do not use the random variable generator in our simulation study since the average of velocities yielded from the random variable generator become smaller than the NGSIM data because of overfitting.

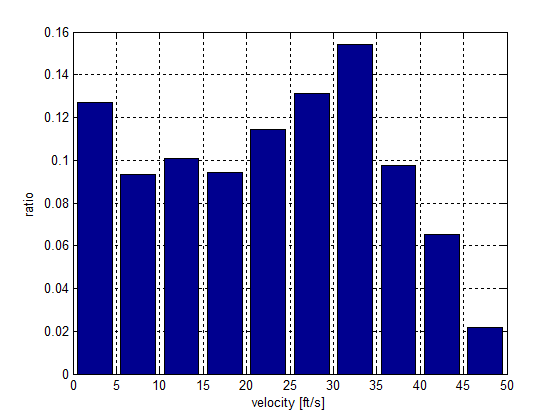


Fig. B3. Distribution of velocity of vehicles from NGSIM data.

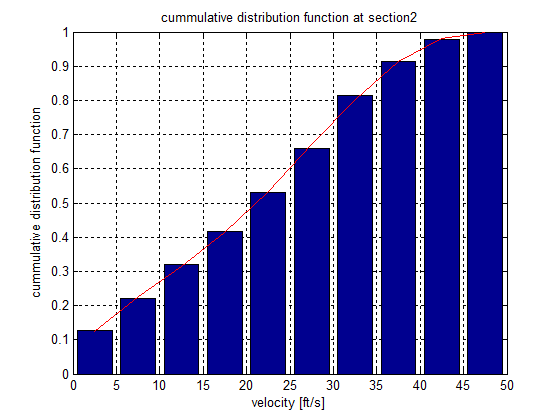


Fig. B4. Cumulative distribution function of velocity