January 31, 2008

RT0759

Computer Science 6 pages

# Research Report

Simulating whole city traffic with millions of multiple vehicle agents

S. Kato, G. Yamamoto, H. Mizuta, H. Tal

IBM Research, Tokyo Research Laboratory
IBM Japan, Ltd.
1623-14 Shimotsuruma, Yamato
Kanagawa 242-8502, Japan

# Simulating Whole City Traffic with Millions of Multiple Driver Agents

**Sei Kato** and **Gaku Yamamoto** and **Hideyuki Mizuta** and **Hideki Tai**
IBM Research, Tokyo Research Laboratory
1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken, 242-8502 Japan
{seikato, yamamoto, e28193,hidekit}@jp.ibm.com

**Abstract**

Simulating large city traffic in microscopic is one of the fundamental technology to understand the nature of urban traffic. Also, nowadays, it is recognized that the variousness of driver behaviors plays an important role in overall traffic. In order to provide an experimental environment to simulate large city traffic with multiple driver behaviors, we designed and developed a large-scale microscopic traffic simulator that simulates traffic of millions of vehicles. We adopt multi-agent simulation framework and have developed the traffic simulator based on a Java-based massive agent-based simulation platform, ZASE. We have so designed the simulator that users can execute traffic simulation by varying driver models and by varying the mixture ratio of the driver models. We apply the traffic simulator to two road networks, a bottleneck road and a road network of the Kyoto city. Our agent-based traffic simulator is found to reproduce the fundamental characteristics of traffic flow, the $q$-$\rho$ fundamental relation with metastable state. We carry out a simulation of the Kyoto-city road network of 32 thousand links and 22 thousand nodes with 0.89 millions of vehicles. In this paper, we clarify the design of the large-scale agent-based traffic simulator and discuss simulation results.

## Introduction

### Motivation

Automobile traffic causes a wide range of traffic problems, such as traffic congestion problem, traffic safety problem, city's air pollution, noise problems, and global warming issue. These traffic problems in metropolis are major problems in today's modern society. Solving these traffic problems in simulation attracts more attention since simulation requires low cost and is free from risk comparing pilot programs. A significant amount of work has been done to model and to simulate traffic since the pioneering work by Lighthill-Whitham and Richards in 1950s (Lighthill & Whitham 1955; Richards 1956). In 1990s, studies in modeling traffic from the viewpoint of the many-body interacting system has been developed rapidly (see for example (Chowdhury, Santen, & Schandshneider 2000; Helbing 2001)). Inspired by these microscopic traffic models, today, lots of multi-agent traffic simulator which models human driving behavior applying multi-agent simulation technologies (Dresner & Stone 2005; Yamashita *et al.* 2005; Cetin, Burri, & Nagel 2003; Cetin & Nagel 2002). A common issues in these prior works are two fold. Firstly these multi-agent simulators adopt some macroscopic models in calculating the vehicle speed such as queue model and cellular automaton model. Thus these simulators are not completely "agent-based". Secondly, the number of vehicles in a traffic system generated in these models are up to 100 thousands, which is insufficient to simulate traffic in metropolitan city such as New York, London, and Tokyo, where 2 to 4 million cars are owned.

Recently Yamamoto *et al.* has developed a Java-based massive agent-based simulation platform named "ZASE" (Yamamoto, Tai, & Mizuta 2006). By the use of the thread pooling technology, ZASE enables to host over millions of agents on a single personal computer. Thus, ZASE provides a running environment for users to execute massive agent-based simulation such as auction simulation, mega-scale human navigation, and artificial market. Applying ZASE to large-scale traffic simulation is straightforward and thus it is expected that ZASE realize a large-scale agent-based traffic simulator. ZASE not only provides basic functions to host over millions of agents, but also provides capabilities for massive agent-based simulation applications to run on multiple computers connected with a high performance network. By using ZASE on a PC cluster, we expect that we can simulate further higher number of vehicles.

In this paper, we propose a new traffic simulator for simulating the metropolitan-area scale traffic in microscopic, which we refer to as the "Metropolitan-Area Scale Microscopic Traffic Simulator" (MASMITS). The key property of MASMITS is that it deals the largest-scale traffic with millions of driver agents, which has become possible powered by ZASE. Also MASMITS is the first traffic simulator to simulate all driver behavior with the granularity of vehicles.

### Summary of contributions

This paper contributes to the literature from application side. Our contribution is to offer an experimental environment on which user can execute various kinds of traffic simulations. The simulator would be helpful to predict a traffic when we introduce a road pricing scheme, to predict a future traffic in an aging society with fewer children, to estimate the vehicles' carbon-dioxide emissions in a city.

Our paper is the first to simulate the traffic of millions of vehicles with completely agent-based. All the previous work simulate vehicles up to 100 thousands with incompletely agent-based.

The rest of the paper is organized as follows. We overview MASMITS in the next section and explain how we model the driver behavior in the successive section. In the following section, we provide simulation results of MASMITS in two road network cases. The last section concludes our paper.

## Simulator Design

In this section, we describe the design of the metropolitan-area scale traffic simulator, MASMITS. Firstly in the next subsection, we overview MASMITS. Then we give a description of the architecture in the successive subsection and how MASMITS works in the following subsection.

### Simulator Overview

Our simulator is designed to be an experimental environment for simulating large city traffic with multiple driver behaviors.

The strength of our traffic simulator is three fold. Firstly, the simulator can handle large number of agents, up to million drivers, which enables users to simulate the traffic of metropolitan in microscopic. This enables users to simulate the traffic flow in metropolitan area in microscopic with millions of vehicles on a single PC.

The second point is that the interface between the simulation space and agent space is clearly defined. Users can model and develop their own driver model easily. The simulator space provides current states of traffic and the alignment of roads to driver agents in the agent space. The data includes current speed and positions for vehicles, the distance between cars, the curvature and gradient of road on which the specific vehicle is running. Thus all what driver model implementer to do is to model how each driver react by these provided data. The driver models for each driver are easily imported to the traffic simulator and this enables users to simulate traffic with their own driver model effectively.

The third point is that the simulator is designed as microscopic model, which means that the movements of whole vehicle is decided by the logic of each vehicle agent. This is the main difference of our simulator to the former simulator is that the in our simulator, every driver behavior is modeled in microscopic. The former simulator, for example, models traffic congestion as macro phenomena and route choice of individual drivers as micro behavior (Yamashita *et al.* 2005). In this eclectic approach, the speed of each vehicle is determined by the road parameter independently of drivers intension. While these "rigid" models are effective in verifying the routing approach, these models are inadequate in other applications since they lack of the diversified and complex nature of traffic.

### Simulator Architecture

Figure 1 shows the architecture overview of the MASMITS. MASMITS consists of two spaces, the Agent space and the Simulation Space. The Simulation Space manages the network connectivity of roads, states of every vehicles running on the roads. On the other hand, the Agent Space handles individual drive agents, which have their own driver model.

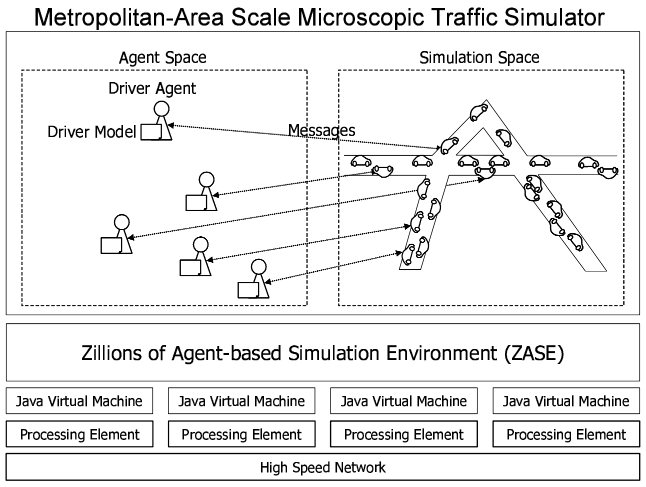

Figure 1: Architecture Overview of the MASMITS.

The interface between these two spaces are clearly defined. From the Agent Space side, each driver agent can see the current states of vehicle, traffic, and roads and each driver agent can find the way to get these data, i.e., Application Programming Interface (API). From the Simulation Space side, each vehicle can find the next speed and next road to go if he or she send a message to the driver agent in the Agent Space.

Since the interface is defined clearly and rigidly, every driver model can be imported into our simulator easily as long as the model is implemented according as the interface of the simulator. One of the interesting challenge is to import driver model which are extracted in a virtual driving simulator (Tanaka *et al.* 2007). At this moment, we have successfully imported these driver models to our traffic simulator and we have succeeded to run the vehicle with this driver model extracted from the real driving simulation experiment.

### Simulation Mechanism

Now let us see how the MASMITS works. As is mentioned in the previous subsection, MASMITS consists of two spaces, the Agent Space and the Simulation Space. In the Simulation Space, time passes clockwise with the default time step $\Delta t = 0.1$ (sec). The Simulation Space roughly consists of three components: vehicles, nodes, and links. Each vehicle preserve its speed, the current position, the current link on which it is running and so on. Each link holds data on vehicles running on its link, the legal speed of the link, horizontal and vertical alignments etc. Location data, that is longitude and latitude is held by each node.

Firstly, when a vehicle starts its origin, the corresponding driver agent is generated in the Agent Space. At every simulation cycle, each driver agent receives the following six types of messages from the simulation space,

- NextSpeedMessage
- EnterSpeedMessage
- NextRoadMessage
- EnterRoadMessage
- ExitRoadMessage
- StartMessage

After making decisions based on its own driver model, the driver agent send back messages to the Simulation Space. Then each vehicle moves as is instructed in the reply message. Finally, when the vehicle reaches its destination, the corresponding driver agent destroyed and removed from the Agent Space. Thus the simulation time evolutes as dumping the log files of every data on each vehicle.

### Road Networks Modeling

In this subsection, we briefly describe the design of road networks. The road network is modeled by the directed graph composed with a set of arcs and a set of nodes. Thus in our simulator, an arc represents a one-way road and traffic lanes are described by multiple arcs. As is mentioned in the previous subsection, each arc holds the legal speed, horizontal and vertical alignments, the length of the arc. Drivers can change their vehicle speed according as the legal speed, horizontal and vertical alignments. Each arc is designed to have a traffic signal on the head of the arc, each of which has three parameters, that are cycle length, offset time, and split time. Every driver agent sets its vehicle speed to zero if the traffic signal turns red.

### Driver Model

We developed a driver model by implementing the algorithm described in the next section using the Java programming language. One of the strength of the object oriented programming (OOP) is the inheritance of properties of the other class. By using this function of the OOP, variety of types of driver agent can be developed with effectively. The class hierarchy for the driver model is organized as follows. The root of the class hierarchy is a Driver class, which is an interface for any driver class. The successive AbstractDriver class is an abstract class for any driver, which implements base driver behavior. Any driver model, such as aged driver, young driver, strategic driver, naive driver can be implemented by extending this AbstractDriver class.

## Modeling Driver Behavior

We examined the essential behavior of driver and summarized the reaction of drivers in the 6 patterns. At this moment, messages for lane changing, over taking and actions for turning right/left are not yet build. Our main message here is not that the driver behavior is well formalized but that the simulator is designed for these additional messages to be added easily. In the following, we will provide the algorithms and the input/output of the driver model. Note that the following algorithms are for base driver model and various driver models are defined by setting different model parameters or by implementing the other logics.

```
$G$      /* gap between tail and nose */
$C$      /* cruise speed */
$V$      /* current speed */
$N$      /*next speed */
$A$      /* acceleration ratio */
$\Delta t$   /* time step size*/
$T$      /* variable */
$P$      /* current position */
$L$      /* length of the road running */
$S$      /* traffic light */
if $V + A \times \Delta t \geq G/\Delta t$ then
  $T = G/\Delta t$
else if $V \leq C$ then
  $T = V + A \times \Delta t$
else
  $T = C$
end if
if $L \leq P + V \times \Delta t$ then
  if $S$ is green then
    $N = T$
  else
    $N = 0.0$
  end if
else
    $N = T$
end if
return $N$
```

Figure 2: NextSpeedMessage Algorithm

### NextSpeedMessage

The NextSpeedMessage returns the next speed for the vehicle to move. Driver agents can select their next speed by taking followings into account,

- distance between vehicle ahead
- current speed
- their own cruise speed
- legal speed of the running road
- gradient of the running road
- curvature radius of the running road

The algorithm for this message is listed in Figure 2. In our driver model, the speed in next simulation cycle is determined mainly comparing the cruise speed and the gap between the vehicle running ahead. If the gap is large enough, the driver agents determine to drive their vehicle at their favorite cruise speed, whereas if the gap is small, the driver agents slow down to an appropriate speed not to bump into a vehicle ahead. In this model, the gap is defined as the distance between the tail of the vehicle ahead and the nose of the vehicle which driver agent is running. In our simulation, the length of each vehicle is taken as 7 meters for every vehicle.

```
N      /* next road */
D      /* direction for the destination */
A      /* azimuthal of the road */
I      /* variable */
T      /* variable */
Let $I$ be infinite.
for each $R$ connected to $CP$ do
  $A$ = azimuthal of the road $R$
  $T = |A - D|$
  if $T \leq I$ then
    $N = R$
    $I = T$
  end if
end do
return $N$
```

Figure 3: NextRoadMessage Algorithm

### EnterSpeedMessage

EnterSpeedMessage returns the speed for the vehicle to enter the next road. Driver agents can select their next speed by taking account the space between the last vehicle in the next road in addition to the traffic and road data previously mentioned. The driver agents determine the speed to enter the next road in a similar manner as the NextSpeedMessage. If there exit a space to enter, which is larger than the vehicle size, i.e.,7 meters, then the driver agents determine to enter at a moderate speed.

### NextRoadMessage

NextRoadMessage returns a road for a vehicle to move in next simulation cycle at a cross point. Driver agents can select which way to go by sensing the direction toward their destination, the density of the next road. Also driver agents can change their route according as the global traffic information if provided. A mount of work has been done for these route choosing algorithm previously. In our driver model, the driver agents select their way that minimize the direction to the goal (Figure 3). A strategic driver model can be defined as to chose the second best link in case the chosen road is in jam.

### EnterRoadMessage

EnterRoadMessage returns a flag for a vehicle whether to enter the next road. Driver agents can select whether to enter or not to enter by taking the space in the next road into account. If there exists enough space for a vehicle to enter in the next road, then the model algorithm returns true. If the space is less than the vehicle size, 7 meters, then the model algorithm returns false.

### ExitRoadMessage

ExitRoadMessage returns for a vehicle whether to exit the road on which the vehicle is currently running. Driver agents judge whether they should exit or not in consideration of the traffic light. The algorithm for this messages is listed in Figure 4. If and only if the position located in the next simulation cycle would be out of the road length and the traffic signal is green, the driver agents determine to exit.

```
P      /* current position */
V      /* current speed */
L      /* length of the road running */
Δt     /* time step size */
B      /* flag */
S      /* traffic light */
if $L \leq P + V \times \Delta t$ and $S$ is green then
  $B$ = true
else
  $B$ = false
end if
return $B$
```

Figure 4: ExitRoadMessage Algorithm

### StartMessage

StartMessage returns for a vehicle whether to start the origin node by taking into account time. Driver agents start at their scheduled time. If current time is his or her own depart time, then the driver agents determine to start the origin.

## Simulation Results

We have implemented previously designed traffic simulator and driver model in the Java programming language and examined the traffic in the three types of road networks. In this section, we show the simulation results and verify our traffic simulator and driver model.

### Bottleneck Road

To verify the behavior of our traffic simulator and of our driver model, we firstly simulated traffic in bottleneck road. The network consists of three nodes Node 1, Node 2, and Node 3 and two links, Link 1 and Link 2. Each node locates at $x = 0$, $x = 990$ and $x = 1000$ in meter, whereas the legal speed for each link is 60 km/h and 40 km/h (Figure 5). As vehicles pass through the Node 2, we expect that they reduce their speed to observe the legal speed, and this may cause a trigger for traffic jam. We calculated traffics on this road from time $t = 0$ to $t = 360$ with three types of driver agents, who drive its vehicle in speed 1.2 times faster, 0.8 times slower, and in speed by observing legal speed. Each vehicle starts Node 1 bound for Node 3 at his own time. In this simulation, each vehicle leaves the origin every one seconds periodically. As is mentioned later, we monitored states of each vehicles running in the monitoring zone $[x_1, x_2]$, where $x_1 = 700$ and $x_2 = 800$.

Figure 6 shows trajectories for vehicles which start their origin at time $0 \leq t \leq 200$. In the figure, the slope of each line denotes the velocity of each vehicle. We see that the slope of the line changes at around $x \sim 990$ in time $t \sim 77$. This means that the velocity of vehicle decreases at this point and time. This is the start of the traffic jam and we see that

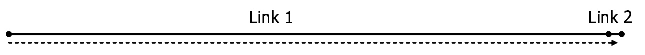

Figure 5: Road network for the bottleneck road. The network consists of two links, Link 1 and Link 2, whose legal speed are 60km/h and 40km/h for each. In the figure, dashed line denotes the trip path for vehicles.

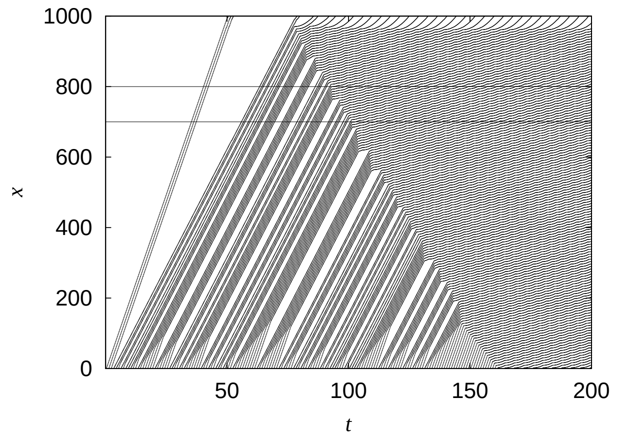

Figure 6: Spatiotemporal plot of vehicles in time period $0 \leq t \leq 200$. Two horizontal solid lines denote for the observation zone. The vertical and horizontal axis represents the space coordinate and the time coordinate, respectively.

the jam propagate backwards until it reaches the origin at time $t \sim 162$. After this moment, the whole road is in jam and the statistics in traffic of this road becomes statistically unstable. The backward propagation speed of the traffic jam is calculated as 22.5 km/h, which agrees with the observed value 20.0 km/h.

To see the traffic characteristics more quantitatively, we calculate traffic density $\rho$ and traffic flow rate $q$. In Figure 7, we show the $q$-$\rho$ relation diagram, which is one of the fundamental and universal characteristics of traffic flow. We observe $q$ and $\rho$ in the Eulerian manner by monitoring incoming vehicles in a observation zone $[x_1, x_2]$. We first observe velocities $v_{ij}$ and the number of traffics $n_i$, which are running on the observation zone for each time $t = i \times \Delta t$ and vehicle $i$, where $x_1 = 700$, $x_2 = 800$, and $\Delta t = 0.1$. The traffic density at simulation cycle $i$ is obtained as $\rho_i = n_i/(x_2 - x_1)$. Then the traffic density $q_i$ at simulation cycle $i$ is calculated as $q_i =< v_i > \rho_i$, where $< v_i >= \sum v_{ij}/n_i$ is a average speed for vehicles $j$ $(1 \leq j \leq n_i)$.

In the figure, we see that in the region where $\rho < 0.06$, the traffic flow rate increases as the traffic density increases, whereas in the region where $\rho > 0.06$, the traffic flow rate decreases as the density decreases. This means that the phase of the traffic flow transits from the free flow sate to the jam state as traffic density increases. We cannot clearly see the critical density $\rho_c$, but it is roughly estimated as $\rho_c \sim 0.060$, which gives 2.4 times larger value than the ob-

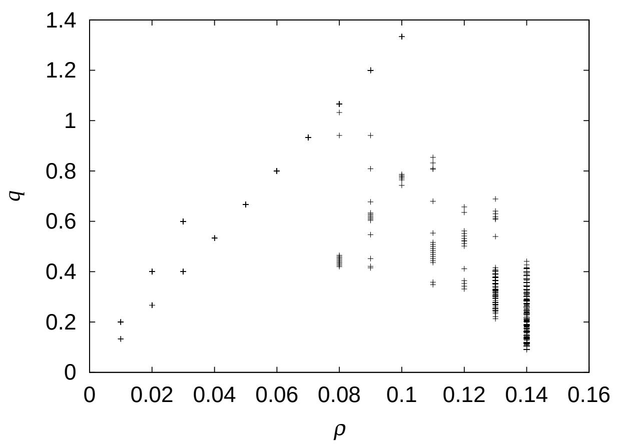

Figure 7: Scatter plot of ($\rho_i$, $q_i$) for each simulation cycle $i$. The vertical axis denotes the traffic flow [vehicles/sec], and the horizontal axis denotes the traffic density [vehicles/m].

served value 0.025. We have not yet clear explanation for this gap, but we believe this is due to the difference of the legal speed limit of the simulation and the real express way.

The noteworthy point in our simulation result is that it shows not only the phase transition but also the overshooting behavior. In the figure, we see that the line in the free flow region stretches toward upper right of the apex of the triangle. This overshooting behavior is called metastable state, which is an essential property of the express way traffic. This simulation result is notable in two points. Firstly this is the first observation of the phase transition with metastable state with the agent-based traffic simulator. Secondly, our driver agent model is so simple to emerge this complex phenomena comparing the previously studied cellular automaton models (Barlovic *et al.* 1998; Nishinari & Takahashi 2000; Takayasu & Takayasu 1993).

## Kyoto City Road Network

To show the effectiveness of our large-scale simulator, we simulated the traffic of the Kyoto city, which is one of the populous city in Japan. We created the road network of the Kyoto city, which consists of 32,654 links and 22,782 nodes. The 894,802 origin and destination pairs are assigned accordingly the person trip data. In this simulation 894,802 vehicle agents start their origins toward destination at the start time. The time step size is 0.1 seconds and we calculated the traffic up to 6 seconds. We use the IBM xSeries 335 (Intel Xeon 2.8GHz processor, 4GB main memory) for the calculation.

In Figure 8, we show the network for the Kyoto city and the location of each vehicle at time 4.1 seconds. Time required for calculation is 180 seconds, which means that the real time ratio (RTR), which describes how much faster than reality the simulation is, is 0.033. Now let us compare our simulator with prior works. As to the maximum number of vehicles, MASMITS gives larger maximum number of vehicles simultaneously running comparing with the prior agent-

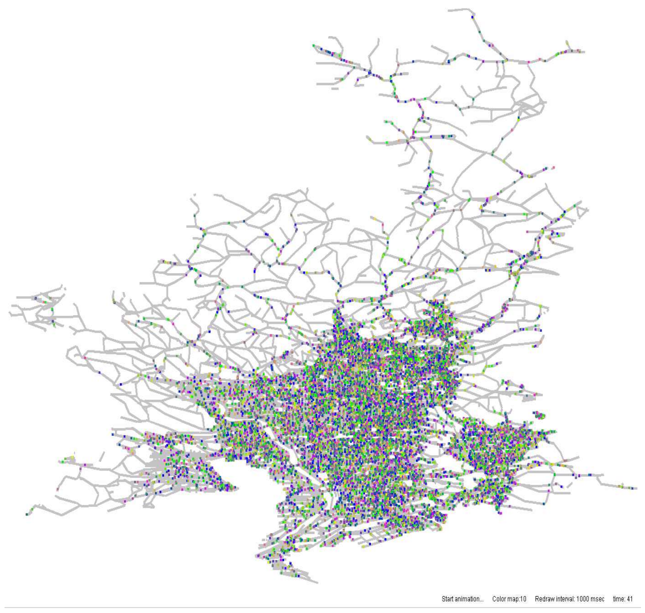

Figure 8: Road network and vehicles for the Kyoto city. Each colored dot denotes a vehicle.

based microscopic traffic simulator (Cetin, Burri, & Nagel 2003; Cetin & Nagel 2002). This queue-model agent-based microscopic traffic simulator can simulate 160,000 vehicles simultaneously with 64 CPUs, whereas MASMITS can simulate 1000,000 vehicles simultaneously with 1 CPU. As to the simulator performance, it is difficult to compare since not only the computational environment but also the simulation model, time step size and the number of messages per time step differs from one simulator to another. The RTR for the queue-model simulator is 190 for the simulation of 160,000 vehicles with 1 second time step, whereas that of MASMITS is 0.033 for the simulation of 890,000 vehicles with 0.1 second time step. The RTR of our simulator is smaller than that of the queue-model. The courteous normalization should be taken to compare these values. Further statistical investigation of the Kyoto city traffic and the performance analysis of our simulator are now under way and will be reported elsewhere.

## Conclusion

In conclusion, we have designed and developed a large-scale agent-based traffic simulator as an experimental environment for large-scale microscopic traffic simulation. On this environment, we can simulate traffic flow with millions of multiple driver agents, which can be imported with ease.

Our simulation results in bottleneck road are found to show the fundamental characteristic of traffic flow, the phase transition with the metastable state. We simulate the traffic of Kyoto city of 32 thousand links and 22 thousand nodes with 0.89 millions of vehicles. The RTR for the simulation with time step size 0.1 seconds is shown to be 0.033.

Performance analysis, validation, and verification of our traffic simulator is now under investigation and will be reported elsewhere. Also we are now analyzing the statistical properties of traffic flow in the Kyoto city road network and will be clarified in the upcoming paper.

## Acknowledgments

The authors would like to thank T. Ishida, H. Hattori, and Y. Nakajima of Kyoto Univ. for their fruitful comments. This work was supported by ”Strategic Information and Communications R&D Promotion Programme (SCOPE)” of the Ministry of Internal Affairs and Communications, Japan.

## References

Barlovic, R.; Santen, L.; Schadschneider, A.; and Schreckenberg, M. 1998. Metastable states in cellular automata for traffic flow. *Eur. Phys. J. B* 5(3):793–800.

Cetin, N., and Nagel, K. 2002. Parallel queue model approach to traffic microsimulations. In *Swiss Transport Research Conference*.

Cetin, N.; Burri, A.; and Nagel, K. 2003. A large-scale agent-based traffic microsimulation based on queue model. In *Swiss Transport Research Conference*.

Chowdhury, D.; Santen, L.; and Schandshneider, A. 2000. Statistical physics of vehicular traffic and some related systems. *Phys. Rep.* 329:199–329.

Dresner, K., and Stone, P. 2005. Multiagent traffic management: an improved intersection control mechanism. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, 471–477.

Helbing, D. 2001. Traffic and related self-driven many particle system. *Rev. Mod. Phys.* 73:1067–1141.

Lighthill, M., and Whitham, G. 1955. Kinematic waves i: Flood movement in long rivers, ii: A theory of traffic flow on long crowded roads. *Proc. Roy. Soc.* A 229:281–345.

Nishinari, K., and Takahashi, D. 2000. Multi-value cellular automaton models and metastable states in a congested phase. *J. Phys. A* 33:7709–7720.

Richards, P. 1956. Shock waves on the highway. *Operations Research* 4:42–51.

Takayasu, M., and Takayasu, H. 1993. 1/f noise in a traffic model. *Fractals* 1(4):860–866.

Tanaka, Y.; Nakajima, Y.; Hattori, H.; and Ishida, T. 2007. A driver modeling methodology for traffic simulation. In *Pacific Rim International Workshop on Multi-Agents (PRIMA-07), Lecture Notes in Artificial Intelligence, Springer-Verlag*.

Yamamoto, G.; Tai, H.; and Mizuta, H. 2006. A platform for massive agent-based simulation and its evaluation. In *Proceedings of the Sixth Intl. Joint Conf. on Autonomous Agents and Multiagent Systems*, 905–907.

Yamashita, T.; Izumi, K.; Kurumatani, K.; and Nakashima, H. 2005. Smooth traffic flow with a cooperative car navigation system. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, 478–485.