Prototype of a Web-based Research-and-Education-Oriented Traffic Simulation Platform

Xuan Shi (Corresponding Author)
Research Assistant
Department of Civil and Environmental Engineering
University of Wisconsin – Madison
(971) 344-0613
shi24@wisc.edu

Jing Jin, Ph.D.
Research Associate
Center for Transportation Research (CTR)
Department of Civil, Architectural, and Environmental Engineering
University of Texas at Austin,
Austin, Texas 78701
(512) 232-3124
jjin@austin.utexas.edu

Yang Cheng, Ph.D.
Assistant Researcher
Department of Civil and Environmental Engineering
University of Wisconsin – Madison
Cheng8@ wisc.edu

Steven T. Parker, Ph.D.
IT Program Manager
Traffic Operations and Safety (TOPS) Laboratory
Department of Civil and Environmental Engineering
University of Wisconsin – Madison
sparker@engr.wisc.edu

Jian Zhang, Ph.D.
Lecturer
School of Transportation
Southeast University
No.2 Si Pai Lou, Nanjing 210096, China
101011599@seu.edu.cn

Bin Ran, Ph.D.
Professor
School of Transportation
Southeast University
No.2 Si Pai Lou, Nanjing 210096, China
and
Civil and Environmental Engineering
University of Wisconsin – Madison
Madison, WI 53706, USA
bran@wisc.edu

Word Count

Abstract:	229
Main text:	4766
Figures (7*250):	1750
Tables (0*250):	0
Total:	6745

ABSTRACT

Traffic simulation utilizes complex models to study complicated traffic problems. Those models include data modeling for objects, behavior models for interaction, and other algorithms such as routing, randomization, variation etc. As no unique model fits every scenario, researchers from transportation community have been investigating novel simulation models continuously. This paper, thereby, presents a prototype of a web-based research-and-education-oriented simulation platform, which helps researchers reduce their workload by providing a compact GIS-T data structure, an intricate mechanism to plug in customized models, a friendly web interface for both configuring and coding, and essential automation procedures.

The simulation platform is established on a service-oriented architecture, with core traffic simulation as a web service. Three modules, SimModel, SimEngine and SimFramework, power the platform in data modeling, simulation flow, and user interfaces, respectively. Its nature of open source and open structure let users easily tailor a complicated simulation program by extending the default essential data and simulation models and plugging in the partitioned simulation flow. If needed, classical simulation models are readily available in a repository. Along with useful automation procedures such as network generation from web GIS application, a comprehensive demonstration for running a typical traffic simulation on the platform is shown programmatically. Corresponding user-friendly web interfaces are presented as well. Further research and future work will focus on completing the prototype in order to become a comprehensive traffic simulation tool.

INTRODUCTION

Traffic simulation is a cost-effective tool to analyze characteristics and evaluate alternatives of transportation problems. The research and development of traffic simulation software has been carried out by both researchers and commercial simulation software companies. As new transportation technologies and innovative simulation models emerge rapidly, simulation software expands remarkably on functionality and model availability with the aid of advanced computer technologies, and plays an more and more important role in transportation for operations, planning, research, and education purposes.

Prevailing simulation software suites are developed based on different platforms. Initially motivated by specialization in one of the subjets such as modeling, customizing, analyzing, animation, etc., they grow, substantially, to incorporate cutting-edge features and reduce drawbacks. Besides the enhancement on usability and visualization, efforts on data preparation and advanced modeling in particular have been made the most. Part of the reason is that distractions from non-related time-consuming work on network preparation, the lack of common traffic simulation data input standard, and requirements for fluent programming skills on advanced usage prevents users focusing on core simulation modeling *(1)*.

Transportation research community has addressed such challenges from different perspectives. Institutional users have been trying to endorse a common data standard of traffic and transportation applications for efficient data exchange *(2)*. Rich pools of instant advanced models and APIs (Application Programming Interface) constitutes the strategy of commercial simulation software to meet customers' needs. Open-source frameworks have been utilized to encourage joint contributions and promote simulation educations *(1)*. Maturing technologies of web-based simulation also starts booming *(3)*. TransCAD has launched a web version, which claims to be the first and only web-based GIS-T (Geographic Information Systems for Transportation) platform *(4)*. TurboTraffic sells traffic simulation service online *(5)*. Early work from institutional researchers has explored an internet-based simulation service, which uses commercial package for core simulation along with a web browser as user interface and display *(6)*. More recently, STREET *(7)*, as a web-based instructional tool for transportation courses, provides modules in traffic simulation, geometric design, demanding modeling, etc. *(8)*. Its merits has been seen in research and education, especially distant learning.

Web-based simulation platforms also make a nature connection with the exploding web GIS applications. Huge amounts of accurate geographic information become publically accessible during the last decade. Simulation software has since been providing compatible data structure to take advantage of such resources, which aid studies on cutting-edge GIS-T for simulation.

We argue that a web-based open-source simulation platform that could plug in interchangeable customized simulation models greatly speed up research and application process. Such platform should provide GIS-T compatible data model that make use of online GIS resources. The simulation models and its data structure should be open and stay compact so that users could build one or more models of their choice with ease and without impacting other models. The plug-in mechanism separates independent components, which stimulates distributed developing. A repository could also be built to encourage model sharing and reusing. Such platform requires less time on learning as a result of the compact models and provides more flexibility on customized modeling. It is an effective platform to build a complicated simulation package rather than an application for running traffic simulation.

This paper presents a prototype of such a platform, which is a preliminary implementation on the previous conceptual design in authors' previous work *(1)*. The prototype materializes the concepts into a solid structure. After exploring the overall structure, it scrutinizes each of the three main modules of the prototype with a focus on design and functionality. The advantageous values of the platform are justified in the presenting of each module. After a demonstration of running a typical traffic simulation, conclusions are drawn.

PLATFORM PROTOTYPE

The platform adopts a high level structure of Service Oriented Architecture (SOA). In the practice of complex development projects, an SOA simulator framework is helpful for an uninterrupted work schedule *(9)*. The mechanism is that simulators are used to simulate the uncompleted modules in the project so that such incomplete modules look complete to the entire project as long as they provide required interfaces and unsurprising outputs. In our proposed platform, the simulator, which offers traffic simulation service, doesn't retire after the project is finished, as it is part of the core of the platform.

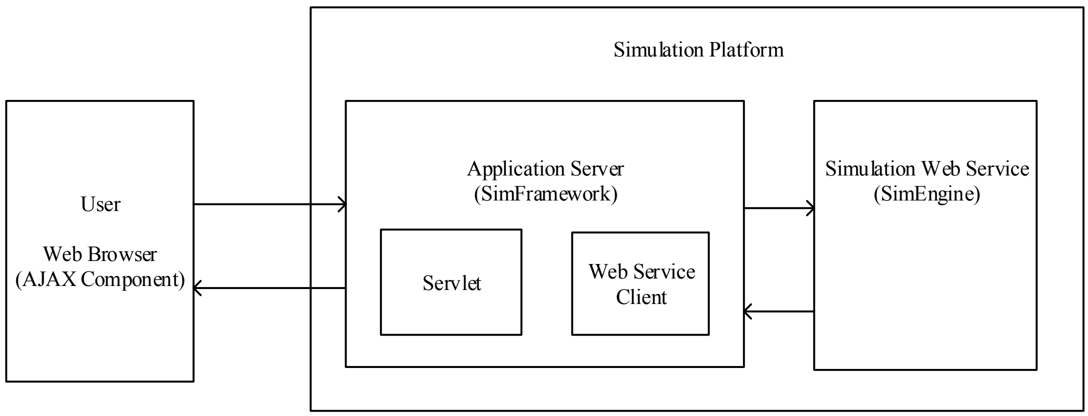


Figure 1 High-level structure of the simulation platform.

In the high level structure, users use web browser to make communications with the simulation platform. The direct user interface of the simulation platform is the application server. Despite serving as a traditional web server that generates responses as per user requests, the application server also performs as a web service client that generates client stub of simulation configuration for traffic simulation service. Traffic simulator, encapsulated in the web service server, contains the key functionality of traffic simulation. Upon receiving simulation results from simulation service, the application server presents them in a human-friendly format to users.

As per the conceptual design *(1)*, such structure reflects all the concepts. Different functionalities of the simulation platform are divided into independent parts. Users are separated from the actual simulation. Researchers are freed from developing a full procedure of simulation. The connections between those parts are the interfaces, which are deemed as the key elements of this platform.

Technology wise, the proposed simulation platform utilizes JavaScript Object Notation (JSON) as the data format for transferring between interfaces. JSON is popular in transferring lightweight structured data over the web. Representational state transfer (RESTful) web service for simulation is chosen to ensure a seamless performance. The reason for both selections is that there tends to be frequent communications and massive data transfers between the user and the platform, or within the platform, due to the nature of a traffic simulation software. Ajax technology one client side is used to maintain rich interactivity between users and the platform.

Under the structure, three modules, which are SimModel, SimEngine, and SimFramework, form the simulation platform. The foundation of traffic simulation resides in SimModel, which accommodates data modeling and traffic modeling. And it is lightly established on open-source GIS libraries. SimEngine, the stage for simulation, implements the flow of traffic simulation, and enjoys the plug-in mechanism for customized models from SimModel. Lastly, SimFramework, built on servlet technology, is the frontend server for user interface. SimModel feeds both SimEngine, SimFramework as data structure as well as data sources.

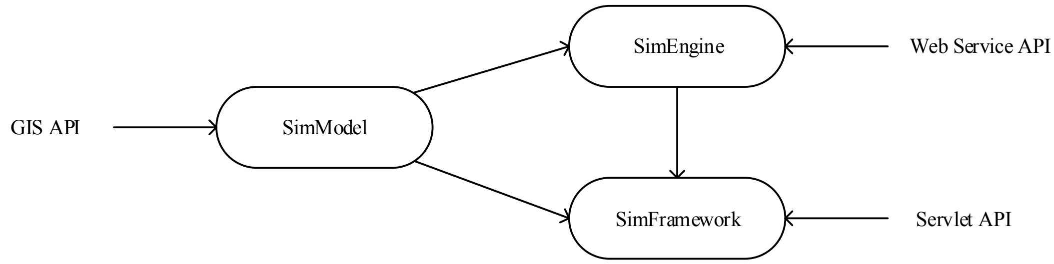


Figure 2 Key components of simulation platform modules.

Modeling - SimModel

As the heart of the traffic simulation platform, not only does SimModel makes all its models customizable and replaceable, but also it provides a flexible data structure that opens up the definition of the data models, which means the customization are not restricted by APIs. As a result, researchers have more freedom in modeling innovative vehicle technologies such as vehicle-to-vehicle communication, cruise control system, etc.

Prevailing traffic simulation packages typically come with a solid data structure. The features of objects are predefined. For example, the acceleration of vehicle and the length of a link are both fixed attributes. Users can only customize their values, or ranges, and, sometimes, tailor an algorithm to calculate them. Despite the relief from defining objects, researchers are often stuck with those interfaces, especially for an unpopular study. To reduce such possibility, simulation packages are growing tremendously in data structure, which tend to cover every corner of the roads, and every detail of the vehicles. And it also comes out in different levels of details (LOD) for different needs. For example, the data structure only cares topology of road network at a regional-level LOD while it provides interfaces for geometry at a street-level LOD. As researches advance quickly and wind suddenly, it is too hard to define a series of universal LOD data structure, or even a lasting one. Thus, rather than offering data structure in different kinds of LODs for researchers to choose from, SimModel provides a compact data structure with only the essential interfaces for basic simulation. Users are allowed and encouraged to implement detailed models. For that reason, the proposed simulation platform doesn't define complicated models, but it collects and shares them.

Data Modeling

In the prototype, SimModel defines four categories of basic objects: 1) fixed objects, 2) moving objects, 3) behavior models, and 4) algorithms. Such grouping stands on the nature of agent-based modeling and simulation for complicated systems. Autonomous agents and their interaction with the environment as well as other agents build the complex system from ground up *(10)*. In basic traffic simulation, the moving objects, primarily vehicles and pedestrians, perform as agents; fixed objects, the road network and infrastructures, function as the environment; behavior models, like car-following models or lane-changing models, power the interaction. In addition, algorithms embody transportation-specific abstract logics.

The approach to represent agent-environment relationship and social interaction are termed as topology *(11)*. It evolves as demand on comprehensive capture of agent and environment soars. Originally, cellular automata (CA) were constructed to relate agents in grid. Later on, more complex Euclidean space, and network topology were introduced to mark the location of the agents *(10)*. Since the explosion of accessible GIS information, the integration of GIS to agent-based simulation attaches agents with geo-spatial information *(11)*. Nevertheless, a systematic technique to effortlessly locate the agents in the environment is the common feature of those approaches. In traffic simulation, such efforts are devoted particularly to a GIS-T road network representation. Classical approaches include node-arc model, linear-referencing model, and fixed / variable segmentation models *(12)*. The idea is to abstract a road network topology into a graph representation with a geographic coordinate system to reference locations. The geometry information of the roads is, however, preserved implicitly in the geographic system. Therefore, agents could be pinpointed to an exact location in the network, given a coordinate of that system. Traditional approach uses a global coordinate system to define the location of nodes and links, and a local coordinate system, with respect to the link, to determine the relative position of a vehicle. The transformation between the global coordinate system and local coordinate system is fairly simple. The advantage of such approach is that the positioning for both road network and vehicle is straightforward with the embedded GIS information.

Geo-spatial information on vehicle and road is indeed essential for ITS *(13)*. The traditional approach abides closely by this rule. However, we argue such approach is great for general transportation but not traffic simulation. First, GIS information is typically heavy. Considering the iterative procedures in simulation, it is inefficient to use it directly and frequently. Second, lane information is hardly included in GIS-T data models. The few lane-based GIS-T models are struggling with the compatibility with other models, especially in routing algorithms *(12)*.

Since the goal of our data structure is essential and compact, we propose a road network data model that stores the geometry information and derives the GIS data, on the contrary. The data model of road network is a combined-link-lane-based model. The model contains the following key elements:

1. Node, point geometry, a coordinate, that denotes the junction point of links.
2. Link, linear geometry that represents the shape of a road.
3. Lane, container for a vehicular queue, and stores the relative position to the centerline of the containing link.
4. Connector, linear geometry that contains lane connection information at a node.

The node and link contain the GIS information and topology information. The playground for vehicles is, instead of link, lane that has few attachments with the link. Lane connects with each other without the knowledge of it parent link so as to provide smooth environment for vehicles moving. The connected link-node graph gives the representation of the road network geographically. As a result, turntables are not necessary. Relative position of a vehicle in a lane only requires 1-D coordinate, which is the traveled distance from the start point of the lane along the link. The geographical location of vehicles can be easily derived from the shift and offset of a lane to the link. Those features are key to the performance of the simulation platform. Figure 3 gives the illustration of the compact data model.

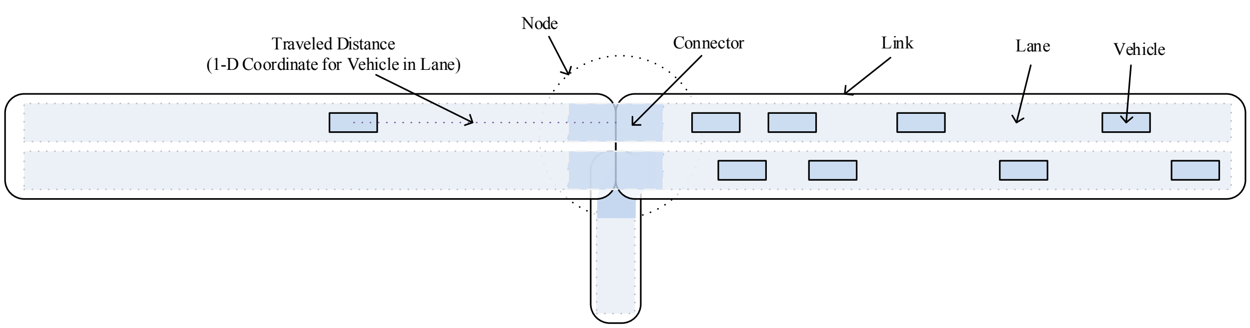


Figure 3 Illustration of proposed compact GIS-T data model.

Implementation

Regarding the compact data model structure, the prototype implemented essential pieces so far. The geometry implementation utilizes open-source geometry library, which are in accordance with specification for simple feature access *(14, 15)*. Both Internet Standard Organization (ISO) and Open Geospatial Consortium (OGC) specify common storage models for geographical data, primarily 2-D geometry. The open nature of OGC specification of different options *(15)* makes it widely adopted by open source community. Major open-source databases support its SQL option of such data structure. SimModel lightly establishes on Java Topology Suite (JTS), which implements OGC simple feature access, for the implementation of the proposed GIS-T data model. To make the geometry coordinates meaningful to GIS, SimModel utilizes EPSG: 900913 *(16)* as the basic coordinate system, which is justified later in the paper. Besides, vehicles provide interfaces for physical and mechanical properties, and a GM car-following model, and turn-ratio routing algorithms are ready for tests.

Despite the compact data model, the true value of SimModel lies in its functionality as both an object manager and a plug-in manager to the whole platform. Objects, used in simulation, are generated, stored, and managed by this module. SimModel provides a clear traffic-simulation-oriented interface; the aforementioned compact data structure is maintained as default implementation. Users have choices to customize certain models by:

1. Decorating the default, which means user adds blocks of procedure with APIs in the default model, and such decorators are only meaningful within the model.
2. Implementing the clear interface, where user could potentially perform complicated modeling.

The plug-in manager monitors user's configuration, and ensure objects linking with the correct implementation. The mechanism behind the plug-in manager is, as Figure 4 illustrates, by matching a type and a plug-in.

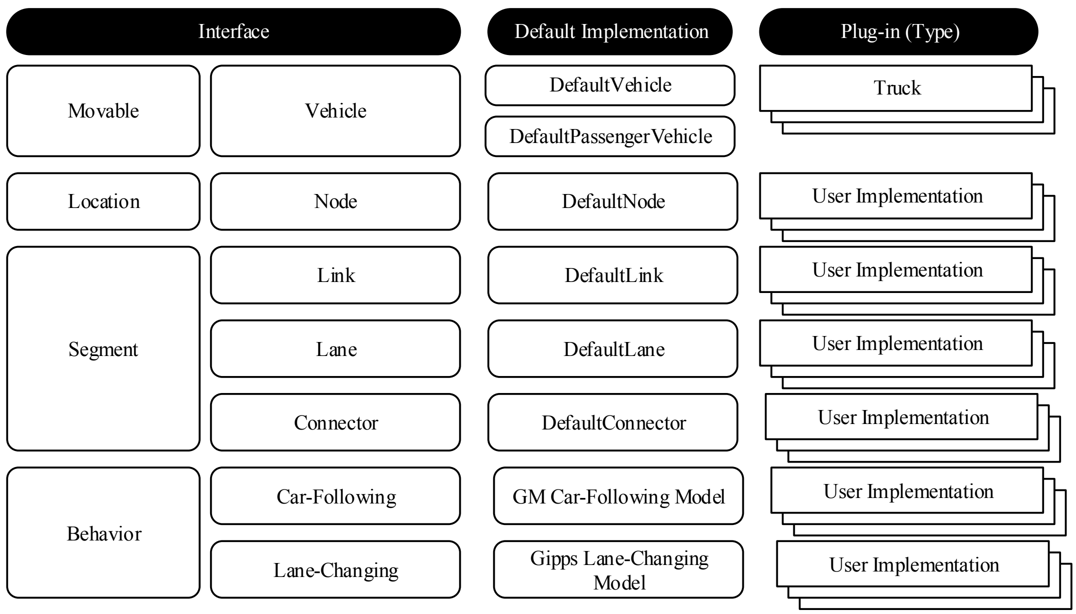


Figure 4 Illustration plug in customized models.

Although compact data structure provides advanced researchers with flexibility, the disadvantage in the lack of options of sophisticated functions makes it less competitive with other strong simulation packages. In time, the proposed simulation platform would collect abundant models, contributed by users. Nonetheless, the availability of well-known classical models enjoys a high priority in our schedule. A comprehensive library of standard vehicles, links, routing algorithms, or car-following models is under serious development, which makes SimModel stay compact in data structure but rich in content. Consequently, such a rich library could be useful for educational purposes as well.

A human-friendly data storing mechanism is still under development. SimModel tends to utilize a spatial database for comprehensive storage, and JSON for light-weighted transfer format. To be a real object manager, SimModel will be responsible for data loading and storing for the whole platform. Its performance affects the efficiency of the whole simulation procedure so that it takes time careful consideration. It is due to be available soon.

In the forthcoming version, SimModel will debut some supplementary objects as well. The modeling for point, linear and moving events, planned or unplanned, traffic rules and restrictions, and traffic signal control draw our key attention, due to their high values to the platform and popularity from the users.

Simulation - SimEngine

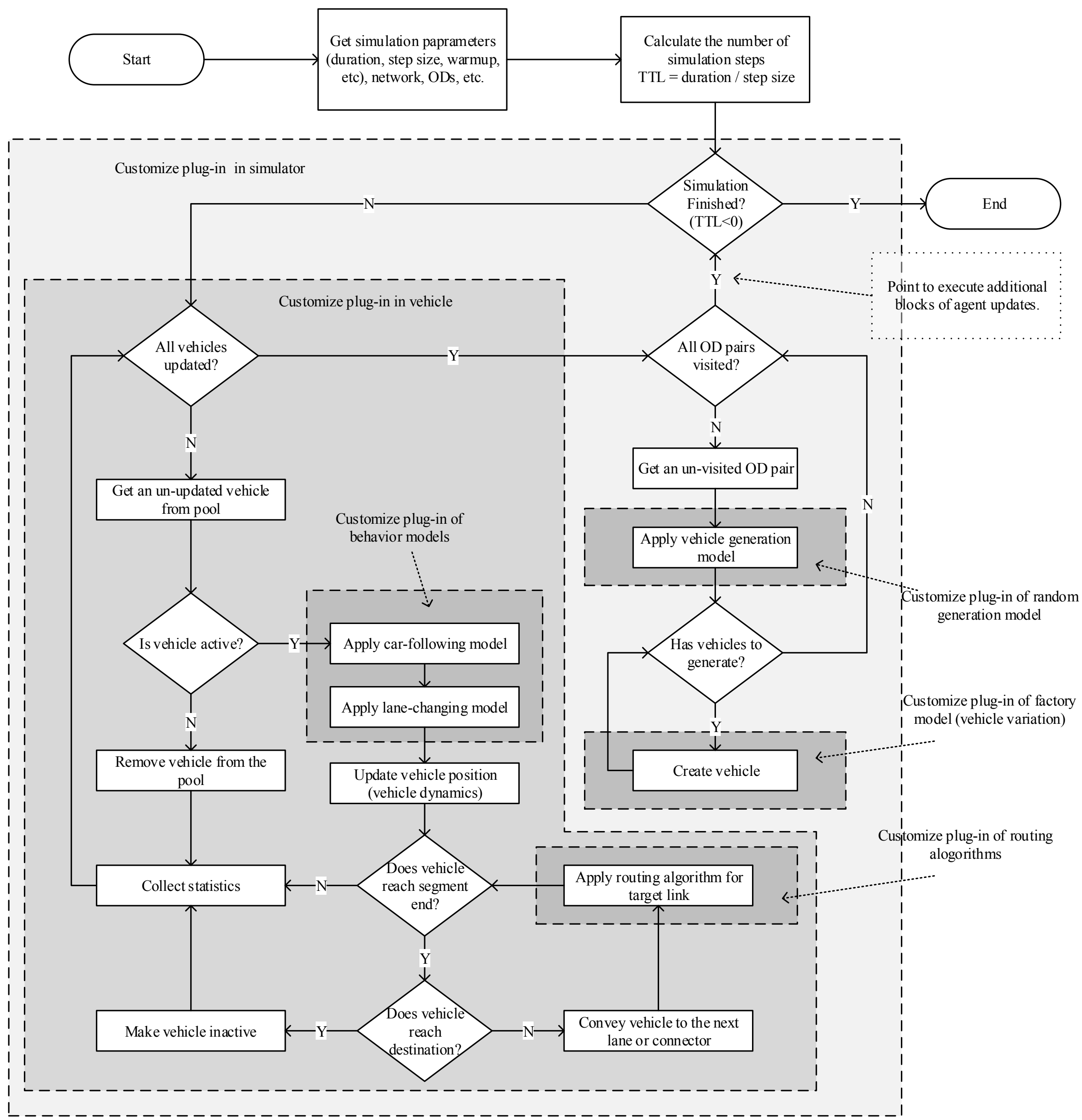
Using the models and objects from SimModel, SimEngine implements the logical flow of microscopic traffic simulation. The flow diagram of SimEngine is as typical as described in *(1, 17)*. However, the prototype provides the same degree of openness as SimModel. The prototype first calculates the total steps of the simulation. And then, within each simulation step, it executes a series of blocks of fundamental procedures, which include updating existing agents, generating new agents, and collecting statistics of agents. Those procedures are customizable via plug-in at different levels, from a specific model to the entire simulation flow. As for now, SimEngine is plug-able at the following interfaces:

1. Simulator, for the overall simulation flow.
2. Vehicle, VehicleType, for the vehicle properties and mechanics.
3. VehicleMover, for the dynamics of moving vehicle in queue.
4. CarFollowingBehavior, LaneChangingBehavior, for the behavior models.
5. Router, for the routing algorithms.
6. VehicleGenerator, for the generation of vehicle types and emerging time points of vehicles.
7. VehicleFactory, for the variation of vehicles of the same types.
8. VehicleStatisticsCollector, for vehicle statistics collection.

The prototype has a default implementation of those interfaces with essential functionalities and a default simulation flow as illustrated in Figure 6. The default routing algorithm is turn-percentage based. The default vehicle generation model is based on arrival rate with Poisson distribution (the popular alternative is based on headway with negative exponential distribution). Insertion point for additional blocks for other agent activities is also reserved as the Figure shows. Blocks for events, traffic rules, signal controls, etc. will be inserted there as well. The corresponding modification of other agents will be updated in a next version of the prototype.

Automatic network generation, which is long desired by researchers, is another key functional part of SimEngine. The expansion of online mapping services makes the automation process achievable. In fact, such efforts, including transforming road network information from GIS databases (18), extracting geometry from GIS satellite map (19), etc., have already been made. Those methods could provide sufficient information that traffic simulation requires. To fit our compact data model, SimEngine is particularly interested in Esri shapefile and Open Street Map.

More and more transportation agencies in US start providing road network in Esri shapefile format. The LINE product of US-Census-Bureau-supported TIGER (Topologically Integrated Geographic Encoding and Referencing) *(20)* online offers road features for all states. Those freely accessible data become valuable for our network generation module. In addition, Open Street Map has a transport layer, the US part of which largely uses TIGER as original data source. As Open Street Map offers a better user interface of navigation, the proposed prototype takes it as our primary data sources. Consequently, the basic coordinate system for the whole platform is settled for EPSG:3857 (also known as EPSG:900913) *(16)*, an identifier for a coordinate system of Spherical Mercator projection *(21)* commonly used in online GIS community. Due to its favorable unit of meters, which could be directly used in traffic simulation, it outstands from the other popular coordinate system, EPSG:4326, whose unit is in degrees.



* Customization for vehicle involves more than one plug-in.

Figure 5 Microscopic Simulation flow diagram of SimEngine.

Open Street Map has two basic features, node and way, which represent point geometry and linear or polygon geometry respectively. Each feature has tags describing it. A road feature is typically registered as a type of polyline, except roundabout as combined polyline and area. It also has distinctive tags such as highway, lane information, speed limit, etc., which make the road features easily identifiable. Way is essentially composed of nodes, which are basically coordinates. SimEngine in the prototype searches those tags, extracts all the road features as coordinates, and forms the roads in our data structure, within a given geographic boundary. The network, generated by the extraction and formation algorithm, is then presented to the user for further configuration. The algorithm will be presented in a separate technical document.

In short, SimEngine offers various complete procedures to run traffic simulation and parse results. With the help of such procedures, customized simulation could be easily constructed in the flexible by substituting or plugging default or user-defined blocks. Those blocks are formed carefully by the functional partition of simulation flow.

Framework - SimFramework

As the name suggests, SimFramework supports the simulation work for user interaction. It uses web technologies to build a simulation development environment that stands between a sophisticated coding-based software package and a simple click-and-run window application. Advantages of both are incorporated into our main graphic user interface (GUI) in web pages. The prototype provides panels in web pages for users to configure parameters such as properties of vehicles and links, OD matrices, vehicle compositions, etc. Those panels are usually summoned upon click of such physical objects shown on the map, thanks to the direct interaction between user and the online map. When working with non-parametric logics such as routing algorithms, vehicular behavior models, or any plug-in as mentioned in previous sections, an online coding-compilation-and-deployment module enables users run and test their own code pieces in the traffic simulation on the fly. In the meantime, SimFramework collects those models, if given permission from the user, into a central repository. A GUI for user to explore and evaluate those models is ready, though the criteria of evaluation are still under research.

From the aspect of technology, SimFramework is built on, besides static web page technologies, Spring Framework, a Java platform application framework. Particularly, the web servlet and RESTful web services module supplies necessary functions to sustain user interaction and produce simulation results and animation.

Demonstration

Here demonstrates the usage of the prototype to run a typical traffic simulation. At development stage, it offers two options for a test run: 1) from web interface, 2) from program interface (advanced). Although web interface is the shining feature to the platform, a programmatic walkthrough of a test simulation run shows our logic more clearly. A typical user runs the simulation in the following steps, each of which corresponds to one or more panel interfaces dispersed in web pages though.

1. Generate network. It extracts and collects essential road features from third party (Open Street Map), and generates a network in the platform's data structure. It also involves the choice of a link implementation, which comes from the choice of network factory, to customize the links. Alternatively, it can load one from a saved network.
2. Define the properties of network. In addition to the embedded GIS information from Open Street Map, it fills in specific properties such as exact lane width, lane connection or restrictions, etc.
3. Define vehicle types and vehicle factories. It defines involved vehicle types, which could be a user definition of a specific vehicle. Vehicle factory defines how vehicle of the same type varies from each other with a known distribution.
4. Define origin-destination. It defines vehicle type composition, vehicle origin-destination pair locations, and vehicle origin-destination volumes.
5. Define routing. It defines how vehicles are routed in the network once born. It involves the choice of a specific routing algorithm.
6. Define vehicle generation model. It defines the choice of random vehicle generation model.
7. Define behavior models. It defines the choice of specific vehicular behavior models and dynamics models.
8. Define statistics collection. It tells the platform what specific kinds of statistics are wanted, and user-defined collectors could be plugged in.
9. Define simulator. It configures the simulation parameters such as step size, duration, warm-up period and so on. And it also defines the choice of a specific simulation flow.
10. Run simulation.
11. Animation. Users can see the animation over the Open Street Map.

12. Statistics. Link statistic, vehicles statistics, vehicular space-time trajectories, and fundamental diagram are available to display.

Figure 6 illustrates the screenshot of the graphic user interface. Panels for the configuration of network, vehicle, and simulator are available. User can set parameters for those objects. Besides, as Figure 6 shows specifically, an online coding and compilation panel is ready for easy programming. As shown, users are required to implement the update method for the customized car-following model to work properly. Figure 7 illustrates the animation with options for a list of background layers.

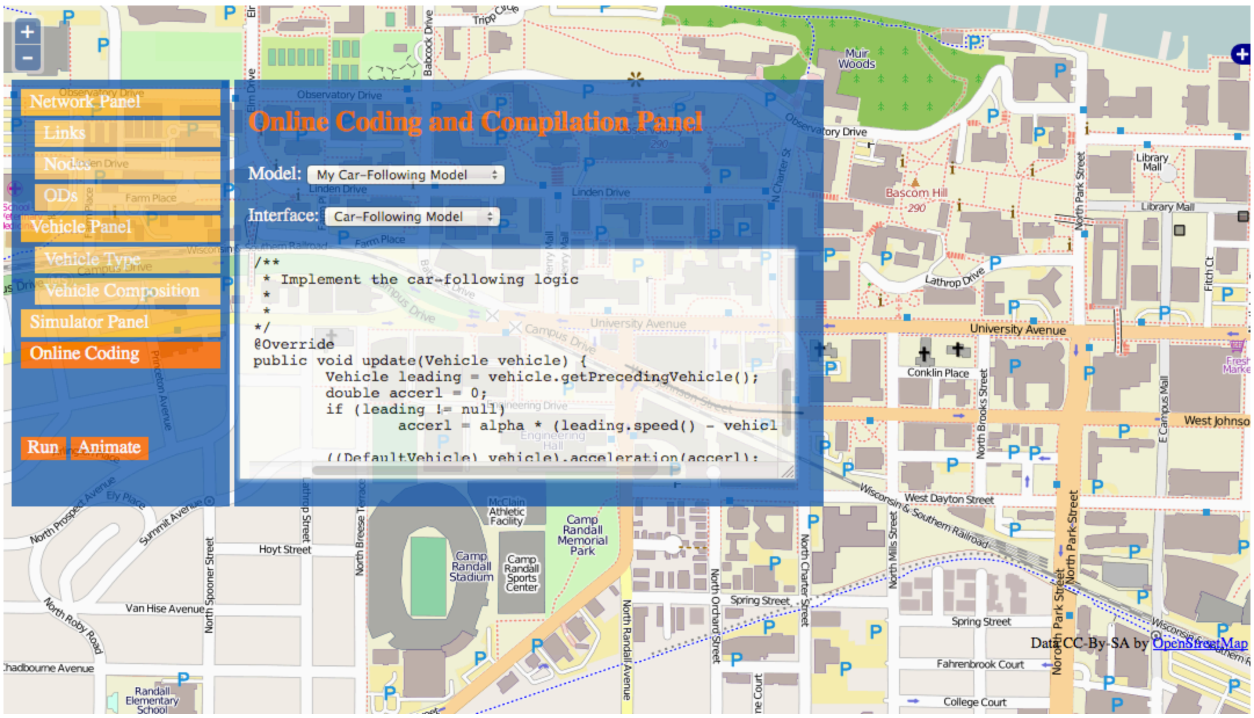


Figure 6 Screenshots of web graphic user interface.

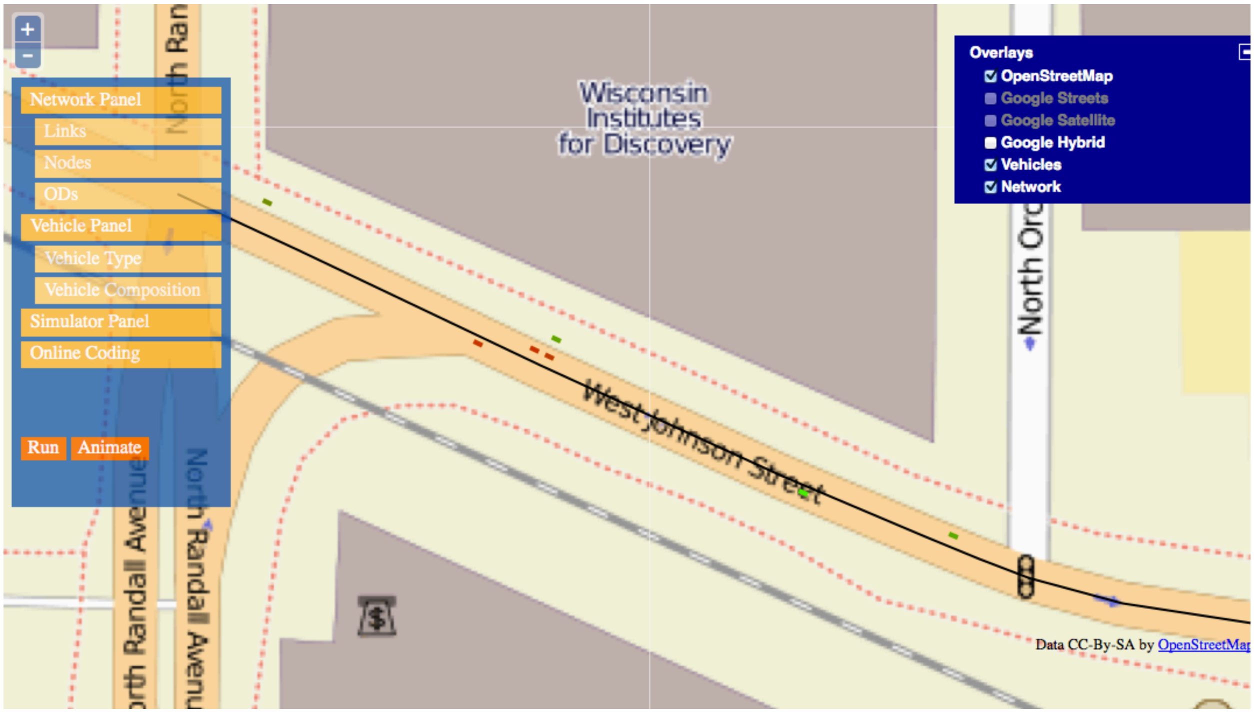


Figure 7 Screenshots of running animation.

CONCLUSION

This paper presents a prototype of a web-based education-and-research-oriented traffic simulation platform. Trusting that no unique model fits every scenario, the platform is established on an open data structure that provides convenient model interchangeability. Those models include popular vehicular behavior models, vehicular dynamics implementation, routing algorithms, vehicle generation models, simulation flow, etc. To support the implementation of complicated models, the platform only offers compact GIS-T compatible data structure, and essential simulation functionalities so that user-customized models could have great programing flexibility and be easily plugged into the simulation flow. Besides, supplementary features are also added to alleviate the dirty works of simulation. Automatic network generation from web GIS applications ensures readiness and accuracy of a road network. A repository, gathering user contributions, spares user with everything customizable in simulation.

Such a platform enhances simulation experience in both research and operational and educational uses. Future work and schedule, as mentioned in each module, are about to transfer the prototype to fully functional simulation software. Instructional implementation of complex models and objects are due to appear on our blueprint. With the justification for the advantages of such an open-data-structure open-source simulation platform, the paper may serve as a guideline to build a flexible GIS-T traffic simulation program.

The development of the proposed simulation prototype into a real platform enters the stage that builds the graphic user interface. When it completes, we intend to conduct a series of applications on the platform, and justify the usefulness of the platform for transportation community. Although the proposed prototype is built for microscopic traffic simulation specifically, the idea and approach are also applicable to establish a general simulation procedure with various interchangeable models. We also plan to integrate macroscopic simulation soon, and resolve some technical issues on absorbing other network problems such as dynamic traffic assignment, route choices, etc. As the development of the platform moves on, our model repository will pile up. As a result, we will use the platform on education for fresh transportation major students as well.

ACKNOWLEDGEMENT

This study is partially supported by the National Key Basic Research Development Program of China (No.2012CB725405).

REFERENCES

1. Shi, X., J. Jin, Y. Cheng, J. Zhang, S.T. Parker, and B. Ran. Conceptual Design for a Research Oriented Web-based Traffic Simulation Platform. Presented at 92nd Annual Meeting of the Transportation Research Board, Washington, D.C., 2013.
2. Tamminga, G., L. van den Brink, H. Van Lint, J. Stoter, and S. Hoogendoorn. Toward GIS-Compliant Data Structures for Traffic and Transportation Models. Presented at 92nd Annual Meeting of the Transportation Research Board, Washington, D.C., 2013.
3. Byrne, J., C. Heavey, P.J. Byrne. A review of Web-based simulation and supporting tools. *Simulation Modelling Practice and Theory*, Elsevier, Vol. 18, No. 3, March 2010, pp. 253–276.
4. TransCAD for the Web Overview. http://www.caliper.com/web/transcad.htm. Accessed Jul. 11th, 2013.
5. TurboTraffic. http://turbotraffic.mak.com. Accessed Jul. 11th, 2013.
6. Liao, C., T. Morris, and M. Donath. Development of an Internet-Based Traffic Simulation Framework for Transportation Education and Training. In *Transportation Research Record: Journal of the Transportation Research Board*, No. 1956, Transportation Research Board of the National Academies, Washington, D.C., 2006, pp. 184 - 192.
7. Simulation Transportation for Realistic Engineering Education and Training. http://street.umn.edu/resources.html Accessed Jul. 11th, 2013.
8. Liao, C., H. Liu, and D.M. Levinson. Simulating Transportation for Realistic Engineering Education and Training. In *Transportation Research Record: Journal of the Transportation Research Board*, No. 2109, Transportation Research Board of the National Academies, Washington, D.C., 2009, pp. 12 - 21.
9. Allegri A., S. Ramanathan, M. Sehome, M. Terry, and M.B. Trevathan. Web service simulator framework solution using Spring. http://www.ibm.com/developerworks/web/library/wa-aj-simulator/index.html. Accessed Jul. 11th 2013.
10. Macal, C.M., and M.J. North. Tutorial on Agent-based Modelling and Simulation. *Journal of Simulation*, Vol. 4, 2010 pp. 151 - 162.
11. Crooks, A.T. and C.J.E. Castle, The Integration of Agent-based Modelling and Geographical Information for Geospatial Simulation. *Agent-based Models of Geographical Systems*, 2012, pp. 219 - 251.
12. Miller, H.J., and S. Shaw. GIS Data Models. *Geographic Information Systems for Transportation: Principles and Applications*. Oxford University Press, 2001.
13. Bespalko, S.J., J.H. Ganter, and M.D. Van Meter. Geospatial Data for ITS. *Converging Infrastructures: Intelligent Transportation and the National Information Infrastructure*. MIT Press, 1996, ISBN 0-262-52215-2.
14. ISO 19125-2:2004 Geographic information -- Simple feature access -- Part 2: SQL option, International Organization for Standardization.
15. OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 2: - SQL option, Open Geospatial Consortium.
16. EPSG:3857 (formerly EPSG:900913). http://wiki.openstreetmap.org/wiki/EPSG:3857. Accessed Jul. 11, 2013.
17. Lieberman, E., and A.K. Rathi. Traffic Simulation, in *Revised Monograph on Traffic Flow Theory*. FHWA, U.S. Department of Transportation, 2012.
18. Wilkie D., J. Sewall, and M.C. Lin. Transforming GIS Data into Functional Road Models for Large-Scale Traffic Simulation. IEEE Transcations on Visualization and Computer Graphics, Vol. 18, No. 6, 2012, pp. 890 – 901.
19. Li, Z., M.V. Chitturi, A.R. Bill, and D.A. Noyce. Automated Identification and Extraction of

Horizontal Curve Information from Geographic Information System Roadway Maps. In *Transportation Research Record: Journal of the Transportation Research Board*, No. 2291, Transportation Research Board of the National Academies, Washington, D.C., 2013, pp. 80 - 92.
20. TIGER. http://www.census.gov/geo/maps-data/data/tiger.html. Accessed Jul. 11th 2013.
21. Spherical Mercator. http://docs.openlayers.org/library/spherical_mercator.html. Accessed Jul. 11th 2013.