

FreeSim – A V2V and V2R Freeway Traffic Simulator

Jeffrey Miller
Department of Computer Science
University of Southern California
Jeffrey.Miller@usc.edu

Ellis Horowitz
Department of Computer Science
University of Southern California
horowitz@usc.edu

Abstract – In this paper we describe FreeSim, which is a fully-customizable macroscopic and microscopic freeway traffic simulator. FreeSim allows for multiple freeway systems to be easily represented and loaded into the simulator as a graph data structure with edge weights determined by the current speeds. Traffic and graph algorithms can be created and executed for the entire freeway system or for individual vehicles, and the traffic data used by the simulator can be user-generated or be converted from real-time data gathered by a transportation organization. The vehicles in FreeSim can communicate with the system monitoring the traffic on the freeways (V2R) or with other vehicles (V2V), which makes FreeSim ideal for ITS simulation and testing of protocol designs. A centralized and a distributed peer-to-peer architecture are both described and supported by FreeSim. FreeSim is licensed under the GNU General Public License, and the source code is available for download from <http://www.freewaysimulator.com>.

I. INTRODUCTION

Simulators provide a convenient way for testing a specific application before deploying it in a live environment. With intelligent transportation systems (ITS), traffic simulators are used to try to determine what the result will be before a specific application is deployed within a transportation system. Traffic simulators can be classified as macroscopic or microscopic, depending on whether they simulate the overall flow of traffic within a transportation system or are focused on individual vehicles within the system. In this paper, we describe an open-source traffic simulator called FreeSim, which provides both macroscopic and microscopic capabilities while supporting vehicle-to-roadway (V2R) and vehicle-to-vehicle (V2V) architectures.

FreeSim is a freeway traffic simulator licensed under the GNU General Public License (GPL) [1]. FreeSim’s graphical user interface (GUI) runs within a web browser using the Adobe Flash plug-in and connects to a Java-based server application via a socket. The Java code uses a MySQL database for any persistent storage needed.

FreeSim allows for different freeway systems to be easily represented by a graph data structure. The data needed to represent the graph of a freeway system in FreeSim are the distances of the freeway segments (edges) and the nodes to which the freeway segments connect (vertices). From the nodes and vertices, the freeway system will be rendered in a

GUI with the amount of time necessary to traverse each freeway segment determined by the distance and the speed of that segment.

FreeSim was created for intelligent transportation systems that gather data from individual vehicles traveling within the freeway system. The vehicles are independent entities that execute as separate threads, communicating with the simulator or other vehicles as they desire. When communicating with a central server, as the vehicles are traversing freeway segments, they can send their current speed and location to a central server. They can also request the shortest path to their destination or the fastest path based on the current speeds. Once they receive the updated path, the vehicles can make their own decisions as to whether or not to change their current path. When communicating with other vehicles, each vehicle will receive data about the speed and location of other vehicles and can make local path routing decisions. This approach decentralizes the path routing and allows individual vehicles to implement their own routing algorithms.

FreeSim provides extensibility by allowing programmers to create their own algorithms to run on the data in the simulator. Shortest path algorithms based on distance and fastest path algorithms based on time can easily be created by implementing two methods. These algorithms can be implemented in the central server for all vehicles to use or in individual vehicles. Six fastest path algorithms are already implemented in FreeSim: Dijkstra’s Algorithm [2], Bellman-Ford’s Algorithm [3,4], Johnson’s Algorithm [5], and the three All-Pairs All-Paths Pre-Computed algorithms [6].

Further, FreeSim allows user-generated data or real data, such as that gathered by a transportation organization, to be used to update the edges in the graph during a simulation period. The progress of individual vehicles or the status of the freeway system as a whole can then be tracked to see how the travel times and congestion change with the updated data.

The remainder of the paper is organized as follows. In section II we provide a description of the related work and other traffic simulators, and section III gives an overview of FreeSim. Section IV describes the centralized and the distributed architectures supported by FreeSim. Section V identifies a number of questions that FreeSim can answer, and the conclusion is given in section VI.

II. RELATED WORK

Many transportation applications currently exist and are still being developed. We have classified seven popular traffic

applications (CORSIM/TSIS [7,8], FreeSim, MITSIM [9], PARAMICS [10], RENAISSANCE [11], VATSIM [12], and VISSIM [13]) into eight different categories: platform, source code availability, cost, transportation network, underlying method of use, vehicle model, data input manner, and data gathering manner. Table 1 provides a summary of this data.

The “Application” field provides a list of the different traffic applications we are comparing with the organization that created the application in parentheses. CORSIM is the only application created for a government organization, which is the United States Federal Highway Administration (FHA). FreeSim, MITSIM, RENAISSANCE, and VATSIM were created by universities, and PARAMICS and VISSIM were created and are supported by private companies. In addition, all of the applications are simulators other than VATSIM, which is the only application designed to run in a driving emulator. Although all of the applications could be used in a driving emulator, the rest of the applications are considered simulators since they are attempting to model a transportation system and show the system based on a certain vehicle model. Maroto, et.al., provide a good overview of other driving emulators, as well as providing another driving emulator with a new driver model in [15].

The “Platform” shows on which operating system the application will run. There is close to an even split between the Windows and Linux applications, and FreeSim allows execution on any operating system since the entire application is written in Flash and Java, which are platform independent.

The “Source Code Availability” is critical for determining how extensible the application is. FreeSim and MITSIM are both open-source, licensed under the GNU GPL [1] and the MITSIMLab Open Source License [14], respectively. The other applications are extensible, but only based on the application programming interface (API) provided by the developers of the software.

The “Cost” data happens to correspond directly with the “Source Code Availability” data, where applications that are open-source are also provided for free, whereas applications that do not provide source code require a fee for licensing.

The “Transportation Networks” field describes the types of roadways that are available in the applications. All of the applications support free-flowing roadways (such as freeways, highways, interstates, etc.) and CORSIM, MITSIM, PARAMICS, VATSIM, and VISSIM all support traffic-regulated roadways, such as roads with traffic signals, stop signs, toll booths, etc. VISSIM also allows railways, bicycle traffic, and pedestrians.

The “Vehicle Model” specifies whether the application shows the transportation system from a bird’s eye view (macroscopic) or as viewed by an individual vehicle (microscopic). RENAISSANCE is the only application that solely models the vehicles macroscopically, and FreeSim and VISSIM both allow for macroscopic and microscopic modeling of vehicles. An overall view of the transportation system can be seen, and an individual vehicle’s progress can be tracked in those simulators.

The “Data Input Manner” describes whether the data used to determine the locations and speeds of the vehicles is done continuously or just by giving their speed at a discrete location and then having the vehicle maintain that speed until the next discrete location in its path. VATSIM simulates a continuous flow since the vehicles in the application change speeds using algorithms based on the vehicles surrounding them coupled with a psychological driver-behavior algorithm. MITSIM and RENAISSANCE use discrete data; however they extend the discrete data to be uniformly continuous through an algorithm such as the traffic state estimator algorithm. For more information on how MITSIM and RENAISSANCE convert discrete data into uniformly continuous data, refer to [9] and [11], respectively. FreeSim allows the data being sent to the vehicles to be continuously updated by the application, and then each individual vehicle can decide whether or not to respond to the received information.

The “Data Gathering Manner” describes how the data is gathered while the simulation is executing. In FreeSim, the individual vehicles communicate their speed and location back to the application or to other vehicles, which is then fed into the algorithms for updating the fastest paths for all of the vehicles. As in the other applications, the data is also gathered in a traditional manner, similar to the way in which transportation organizations use loop detectors, video cameras, or other devices measuring data at discrete locations.

Although each of the applications described have certain advantages over other applications, FreeSim provides three main advantages. First of all, FreeSim is open-source and free to download. Second, the data used to simulate the vehicles in the freeway system can be sent in a continuous or a discrete manner. And finally, which is the main reason we created our own simulator rather than using an existing one, FreeSim allows communication between a central system and each individual vehicle or between vehicles. Much of the research concerning intelligent transportation systems assume that the vehicles are able to communicate autonomously with mobile or stationary devices, and FreeSim, unlike other simulators, has this feature built into the framework.

III. FreeSim OVERVIEW

FreeSim is a traffic simulator that models free-flowing transportation systems as weighted directed graphs. The edges of the graph are the freeway segments that the user would like to monitor, and the nodes are the connections between the segments. There is no limit as to how many edges can emanate from a node, although in most freeway systems this number will not exceed more than about eight (i.e. a 4-freeway interchange like the 4-level interchange in Los Angeles that connects the 5, 10, 60, and 110 freeways). A freeway system is stored in a database with FreeSim, though there is a second program bundled with FreeSim for reading a list of nodes and a list of edges that define a freeway system from text files and populating the database accordingly. Multiple freeway systems can be stored in the same database with unique identifying names.

TABLE 1. SIMULATOR CHARACTERISTICS

Application	Platform	Source Code Availability	Cost	Transportation Networks	Vehicle Model	Data Input Manner	Data Gathering Manner
CORSIM/TSIS (FHWA)	Windows	Closed	Paid	Free-Flowing (FRESIM), Regulated (NETSIM)	Micro	Discrete	Traditional
FreeSim (USC)	Any	Open	Free	Free-Flowing	Macro Micro	Continuous, Discrete	Individual Vehicles, Traditional
MITSIM (MIT)	Linux	Open	Free	Free-Flowing, Regulated	Micro	Discrete Transformed	Traditional
PARAMICS (Quadstone Ltd.)	Linux, Solaris, Windows	Closed	Paid	Free-Flowing, Regulated	Micro	Discrete	Traditional
RENAISSANCE (Technical Univ. de Crete)	Windows	Closed	Paid	Free-Flowing	Macro	Discrete Transformed	Traditional
VATSIM (Ohio State Univ.)	Linux	N/A	N/A	Free-Flowing, Regulated	Micro	Simulated Continuous	Traditional
VISSIM (PTV)	Windows	Closed	Paid	Free-Flowing, Regulated, Railways	Macro (VISUM) Micro	Discrete	Traditional

The graphical user interface for FreeSim renders a freeway system in a browser via the Adobe Flash 8 plug-in (see Figure 1). Over a socket connection, the Flash front-end connects to a Java-based server program for all of the simulator functionality. The Flash interface merely provides a light-weight GUI for displaying the output of the simulator.

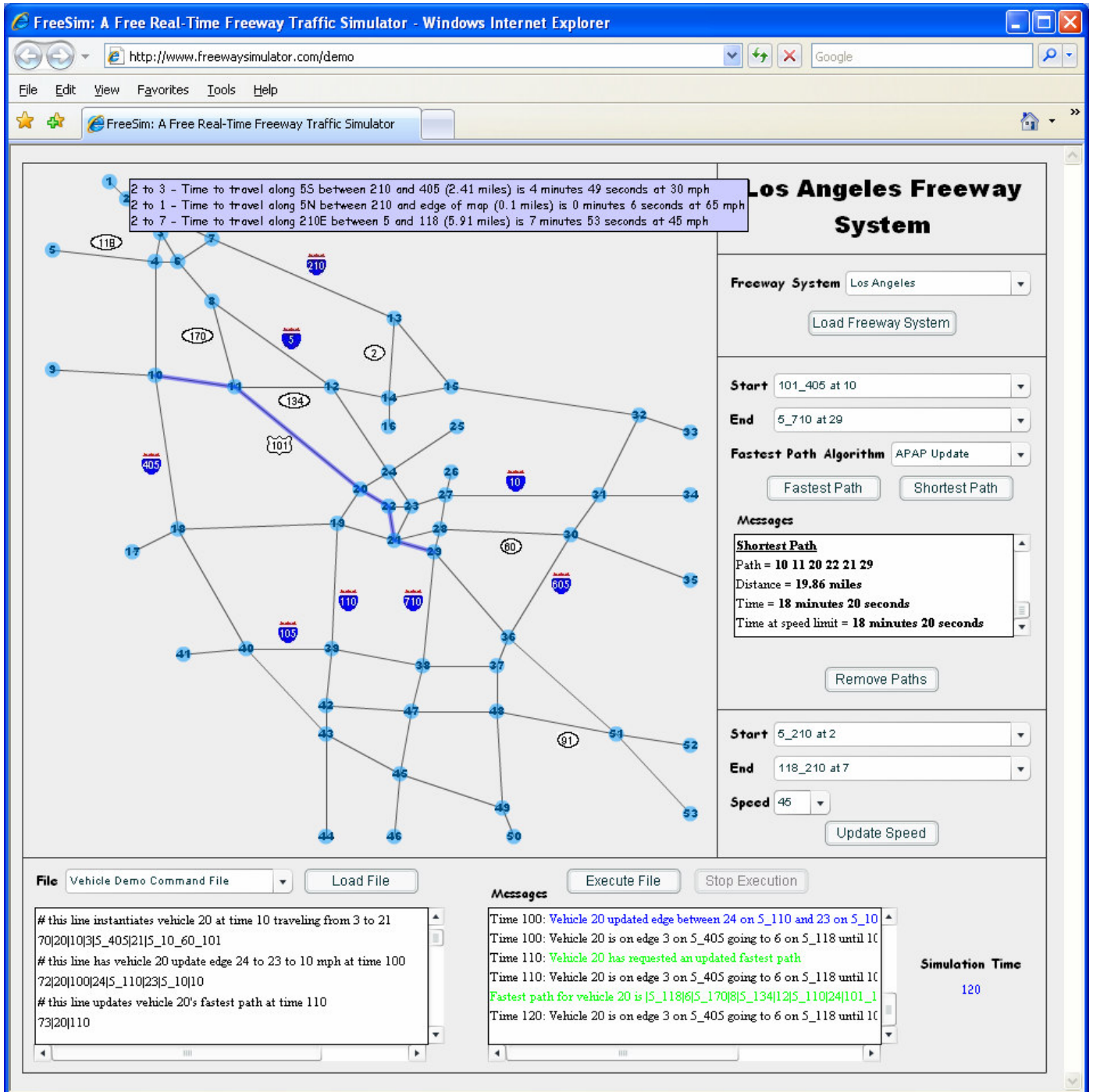
FreeSim allows shortest paths and fastest paths to be determined based on distance or current speeds, respectively. The shortest paths between all pairs of nodes are determined at start-up using Johnson's algorithm, although the fastest path must be determined at query time since it will change based on the current speeds. Fastest path algorithms can be created and executed in FreeSim quite easily by registering a Java class with the traffic simulator that implements two methods – one for retrieving a fastest path from a source node to a destination node, and one for updating the weight of an edge with a new speed. A graph data structure containing the freeway system is also accessible by the class. The method for retrieving the fastest path is called when a user or vehicle requests a fastest path, and the method for updating the weight of an edge with a new speed is called when the speed on an edge is updated to a speed for which the difference is greater than a certain threshold, as defined in [6]. FreeSim already has six fastest path algorithms implemented: Dijkstra's Algorithm [2], Bellman-Ford's Algorithm [3,4], Johnson's Algorithm [5], and the three All-Pairs All-Paths Pre-Computed algorithms [6]. Dijkstra's and Bellman-Ford's algorithms do not have any action associated with an edge update, but rather re-compute the fastest path from the source node when a query is made. Johnson's algorithm re-computes the fastest paths between all pairs of nodes when an edge update occurs, which allows an

efficient fastest path retrieval upon a query. The All-Pairs All-Paths algorithms are more suited for dynamic edge weights, where one of the algorithms executes with a constant edge update time, one with a constant query time, and one that combines the previous two algorithms. The efficiency of these algorithms lies in the static nature of the freeway graph, where the only changing factor is the weights of the edges. Freeway sections are not added or removed very frequently.

In the FreeSim interface, the speed on an edge can be updated, which causes the method for updating the weight of an edge to be called for all of the fastest path algorithms that have been registered with the traffic simulator. The speed and time to traverse each edge at that speed can be viewed by placing the mouse on the top of a node. Those speeds and times will change in real-time as the simulator receives speed updates from vehicles.

In addition to the user being able to find shortest and fastest paths and update the speed on a freeway segment, FreeSim also allows a user to have a series of commands executed simultaneously during a simulation. A file can be created that allows a user to specify different events that will occur at discrete times during the course of a simulation. Among these events are vehicles being inserted into the freeway system at a certain source node bound for a destination node, vehicles sending updated speeds for a specific edge, vehicles requesting a new fastest path based on the current speeds, and messages that allow a user to track the progress of a vehicle along its path and see the messages this vehicle is sending to other vehicles. Since the vehicles are autonomous entities, they also can change their paths or destinations while traveling in the system.

FIGURE 1. SCREENSHOT OF FreeSim GUI



One possible use of this input file is to load the freeway system with live speed data, such as that gathered by a transportation organization. The data obtained by loop detectors can be used to obtain the speed of the vehicles crossing that detector during a given time period [16, 17], which can then be used to create the input file to be executed by FreeSim. This gives FreeSim the ability to simulate the traffic within a freeway system based on live data. With this approach, FreeSim does not generate any traffic data of its

own, which removes the concern that it does not accurately simulate real traffic within a freeway system.

IV. DIFFERENT ARCHITECTURES

FreeSim supports both V2R and V2V communication. When communicating with the roadway, FreeSim uses a centralized architecture, such as that shown in Figure 2. When vehicles are communicating with each other, FreeSim uses a distributed peer-to-peer architecture, such as that shown in

FIGURE 2. FreeSim CENTRALIZED ARCHITECTURE

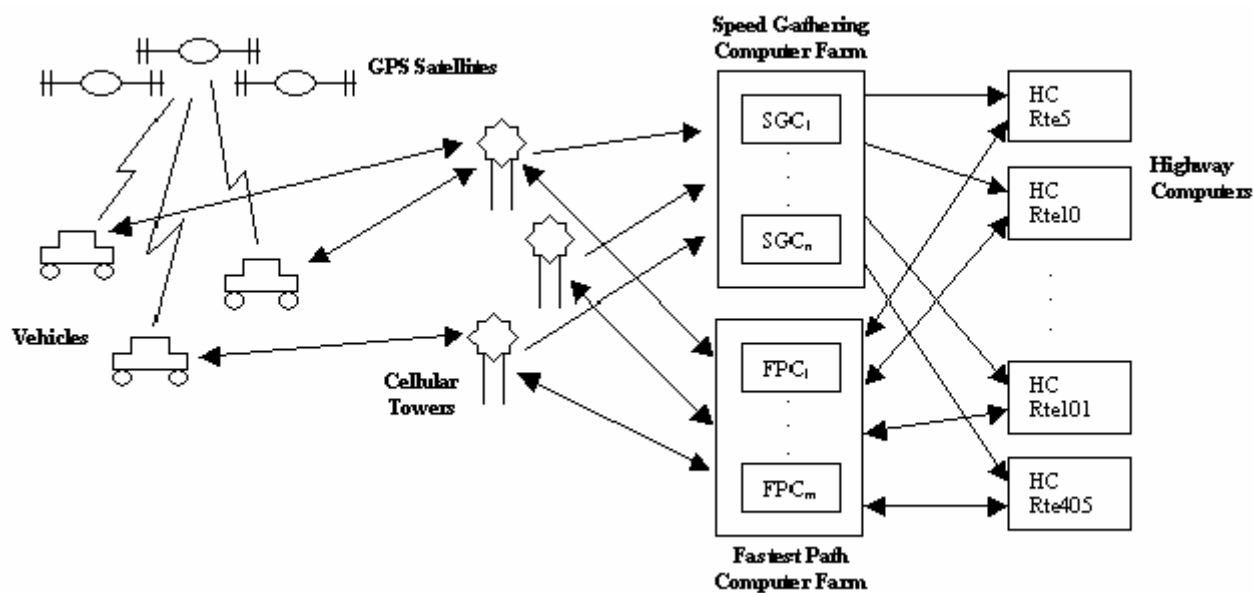


Figure 3. In both architectures, the vehicles are still operating as autonomous entities, so they can make their own routing decisions and choose what data to transmit to the roadway or other vehicles. They can also choose whether or not to take the advice they have received and change their current path.

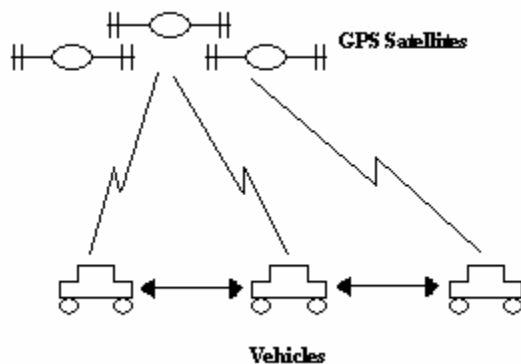
FreeSim's centralized architecture leverages the multi-billion dollar cellular network that already exists along most highways in the world. The architecture assumes that each vehicle is capable of retrieving its geographical location from a Global Positioning System, which is represented as the GPS Satellites in the diagram. For updating a vehicle's speed in the system, the vehicle will send its location and speed (which can be retrieved from the vehicle's computer system or by triangulating a delta in distance during an elapsed time) via cellular towers to the Speed Gathering Computer Farm. The Speed Gathering computers will receive as input a (latitude, longitude) pair and the speed of the vehicle at that location. No output is required, as the vehicle is not requesting any information. The Speed Gathering computers will then map the (latitude, longitude) pair to a highway and forward the location and speed to the corresponding Highway Computer (i.e. HC Rte 10). The Highway Computer will store the data for retrieval by a Fastest Path computer.

For retrieving a fastest path in the FreeSim centralized architecture, each vehicle still needs to be capable of retrieving its location from a Global Positioning System. The driver will enter the desired destination, which will be transmitted along with the current location via a cellular link through a cellular tower to the Fastest Path Computer Farm. The Fastest Path computers will take as input the source location and the destination location. These locations will be mapped to locations on the highway to allow the paths between those two points in the highway graph to be determined. The Fastest Path computers will then request the speed information for each edge in the paths from the corresponding Highway Computer, which will be used to determine the fastest path to be returned to the user.

The FreeSim distributed peer-to-peer architecture will work similarly to the centralized architecture, though the Fastest Path and Highway Computers will exist virtually within each vehicle. When a vehicle is ready to send an updated speed, it will first map its location to an edge of the freeway graph. Only if the speed is different than the speed currently stored for that edge will the vehicle transmit its updated speed to its peers. The peers will average the speed transmitted with their own speed (if the speed is for the same edge they are traversing), and retransmit the speed to all peers other than the one that sent it to them. For all of the vehicles that are not traversing that edge, they will merely update the speed for that edge in their local database. Each vehicle will then be able to make a local decision about fastest paths. There is no need to transmit fastest path information to any other vehicle, and no fastest path queries need to be transmitted.

The centralized approach has a major advantage over the distributed approach in that the entire freeway system can be monitored by an application. In the distributed approach, there is no central process monitoring all of the traffic in the freeway system, but each vehicle has a pseudo-snapshot of the entire system. In addition, there is no historical data in the distributed network, since the data is constantly being updated as new vehicles enter or other vehicles exit the freeway. The centralized approach will be able to maintain historical data for further analysis and trending, whereas the distributed approach is strictly concerned with real-time monitoring of traffic data. However, the distributed approach does not require additional hardware to be installed and monitored and does not have the limitation of a single point-of-failure. Since the vehicles are transmitting the data only to their neighbors, there will be a propagation delay before vehicles can make informed routing decisions. Figure 4 shows the amount of time to traverse what each approach believes is the fastest path at the different times via FreeSim for a path in Los Angeles from the 5/405 interchange to downtown Los Angeles. The

FIGURE 3. FreeSim DISTRIBUTED PEER-TO-PEER ARCHITECTURE



distributed architecture is very jittery as the data is being propagated through the vehicles, whereas the centralized approach is able to more quickly and accurately produce the fastest path. The distributed approach finds faster paths than the centralized approach occasionally if a vehicle has not yet transmitted its own speed to the central server but is using its updated speed in the calculation of the fastest path. As soon as its speed is transmitted in the centralized architecture, the central server updates the fastest path accordingly.

V2R and V2V architectures both have advantages and disadvantages, and so both the centralized and distributed approaches are supported by FreeSim. Traffic and graph algorithms can be executed within FreeSim's framework against live or user-generated traffic data. Based on having an accurate simulation of real traffic data, there are many potential applications and research questions that can be answered.

V. FreeSim SIMULATIONS

FreeSim has been executed with user-generated data and live data gathered from the California Department of Transportation (CalTrans). Any research based on vehicle travel times would be relatively easy to conduct within FreeSim, such as determining the optimal time of the day to travel based on the speeds or occupancy of the freeway system or observing the change in time to travel based on the time of the day at which a vehicle departs from a location. Determining the optimal speed of vehicles to provide the maximum throughput for a specific freeway system could also be measured, as was reported in [20] for the Los Angeles freeway system between midnight and noon. However, during rush hour (between the hours of 7:00a.m. and 10:00a.m. on weekdays), FreeSim has shown that the maximum throughput occurs at both 30 mph (~48 km/h) and 65 mph (~105 km/h), with 21 vehicles passing a point within a 30-second period at both speeds (see Figure 5). This is due to the fact that drivers feel safer driving closer to each other at slower speeds, so more vehicles are able to pass a specific point. At a time that experiences maximum throughput, at 30 mph, each vehicle will have 44 feet (~13 meters) between itself and the vehicle in front of it; at 65 mph, each vehicle will have 118 feet (~36 meters) between itself and the vehicle in front of it.

In addition, any research concerned with the distance traveled during a certain time frame based on changing speeds could be determined. Using the amount of time a vehicle is traveling with the distance traveled during a certain time frame, coupled with average gas mileage of vehicles and price per gallon of gas, the number of gallons of fuel and the amount of money that could have been saved by avoiding traffic could be calculated. All of this data can be used to aid in improving the public transportation systems of a city by providing alternate routes during times of heavy traffic or by giving the systems more accurate arrival and departure times.

Based on the live data gathered by a transportation organization, the added traffic due to an incident (such as a collision, road construction, etc.) can be determined. Alternatively, as an event that attracts a significant number of vehicles begins or ends (such as a sporting event), the effect on the traffic of a freeway system of inserting or removing that many vehicles at a certain node can be measured. Similarly, if a new building were proposed to be built that would attract a number of vehicles each day during a certain time interval, the impact of the added traffic within a freeway system could be simulated. As new freeways are proposed, the overall improvement in the flow of traffic in the entire freeway system could be measured.

From a more theoretical standpoint, graph and traffic algorithms can be tested on user-generated or real traffic data. Shortest path algorithms with changing edge weights (also known as dynamic shortest path algorithms) can be implemented through an API in FreeSim, either at the server for the centralized architecture or in each vehicle for the distributed architecture. Traffic prediction algorithms, such as those in [18] and [19], can not only be tested, but can also be verified against live traffic data. Incident identification algorithms can be executed to determine how long it takes to identify an incident based on real traffic data.

Focusing more on ITS technology, figuring out how many (or what percentage of) vehicles need to transmit their speed and location to a central traffic server or to other vehicles to be able to accurately route vehicles along fastest paths can be determined. With individual vehicles having the ability to communicate as autonomous entities, aggregation algorithms or peer-to-peer approaches can be implemented among the vehicle objects. Having more granular information such as

FIGURE 4. V2V FASTEST PATH VS V2R FASTEST PATH FROM 5S AT 405 TO DOWNTOWN LOS ANGELES ON 2006-11-03 FROM 7:00A.M.-10:00A.M.

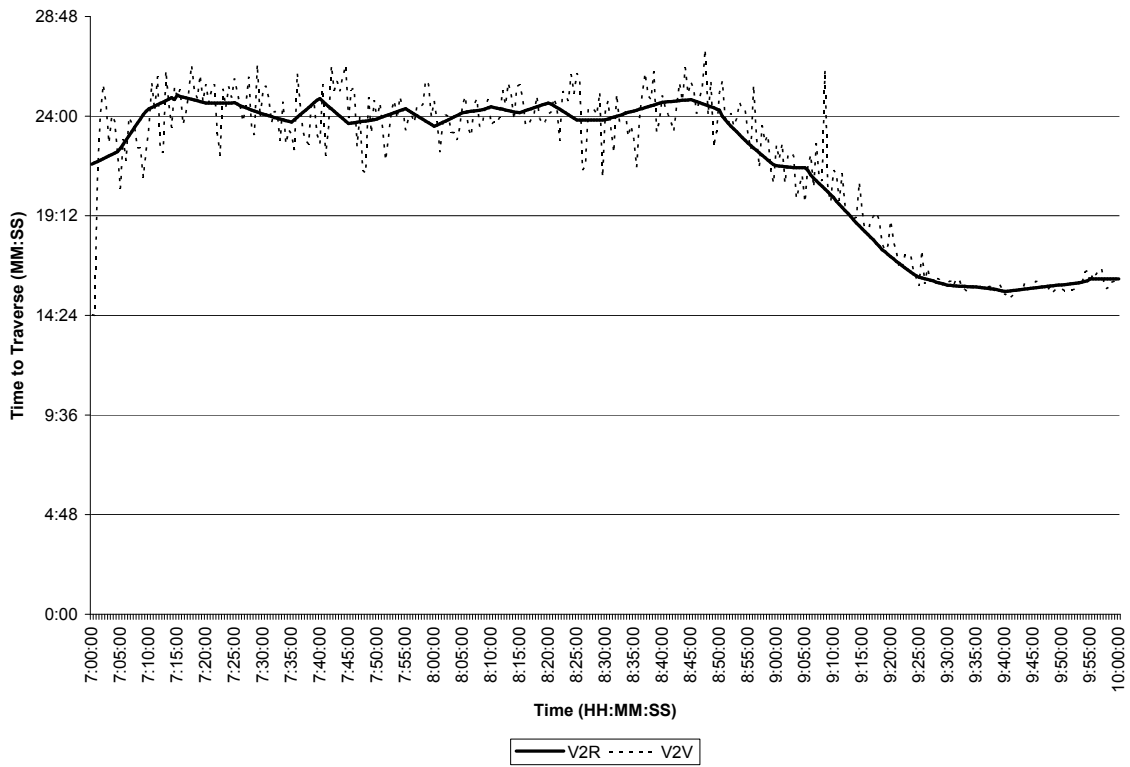
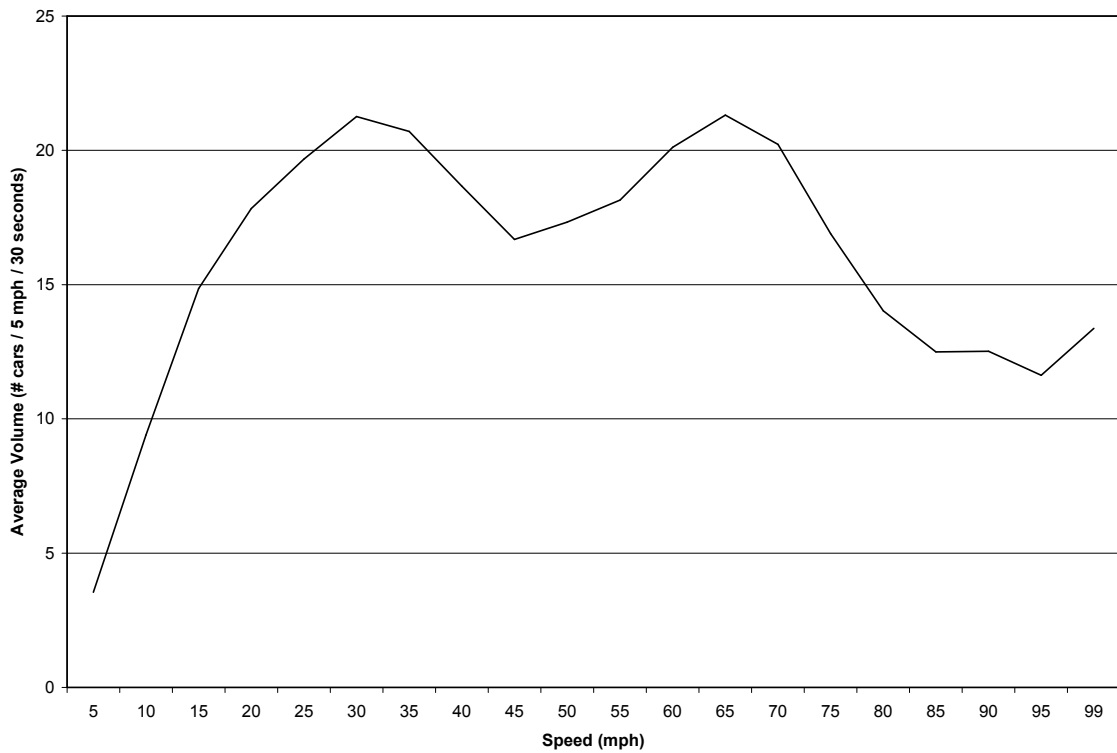


FIGURE 5. VOLUME OF VEHICLES DURING RUSH HOUR (BETWEEN 7:00A.M. AND 10:00A.M.) IN LOS ANGELES FREEWAY SYSTEM ON 2006-11-03



speeds of vehicles in specific lanes gives the ability to determine if lanes are traveling faster than other lanes and therefore saving time for those vehicles.

Although we can not possibly enumerate all of the questions that can be answered using a traffic simulator, this at least provides a few of the open research questions that can be solved using FreeSim.

VI. CONCLUSION

In this paper, we have described FreeSim, which is an open-source simulator that can be downloaded for free from <http://www.freewaysimulator.com>. It offers many advantages over other simulators, with one such advantage being the ability to have vehicles autonomously travel within a transportation system while remaining in constant communication with a central system or other vehicles. With many intelligent transportation systems assuming that vehicle speed and location data will be available, FreeSim provides an ideal test-bed for ITS applications.

FreeSim allows for real traffic data, such as that gathered by transportation organizations, to be used within a simulation, which enables more accurate and credible analyses of traffic scenarios. Two results presented in this paper concern the relationship between V2V and V2R fastest path convergence and the highest throughput for traffic data during rush hour in Los Angeles. The GUI provided with FreeSim is platform-independent and can be run within a browser. Although there are many traffic simulators available, FreeSim is free, open-source (offered under the GNU GPL), fully-extensible, operates on real traffic data, and provides the necessary framework for current and future ITS applications.

VII. REFERENCES

- [1] GNU General Public License. <http://www.gnu.org/licenses/gpl.html>.
- [2] Dijkstra, E.W. "A note on two problems in connexion with graphs." *Numerische Mathematik*, 1959.
- [3] Bellman, Richard. "On a Routing Problem", *Quarterly of Applied Mathematics*. Volume 16, Issue 1, 1958.
- [4] Ford Jr., Lester R., D.R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.
- [5] Johnson, Donald. "Efficient Algorithms for Shortest Paths in Sparse Networks." *Journal of the ACM*, 1977.
- [6] Miller, Jeffrey, Ellis Horowitz. "Algorithms for Real-Time Gathering and Analysis of Continuous-Flow Traffic Data." *IEEE 9th International Intelligent Transportation Systems Conference*, September 17-20, 2006.
- [7] Corsim. <http://ops.fhwa.dot.gov/trafficanalysisistools/corsim.htm>
- [8] Owen, Larry E., Yunlong Zhang, Lei Rao, Gene McHale. "Traffic Flow Simulation Using CORSIM." *Proceedings of the 2000 Winter Simulation Conference*, December 10-13, 2000.
- [9] Yang, Qi, Haris N. Koutsopoulos. "A Microscopic Traffic Simulator for Evaluation of Dynamic Traffic Management Systems." *Transpn Res., Part C*, Volume 4, Number 3, 1996.
- [10] Cameron, Gordon, Brian J.N. Wylie, David McArthur. "Paramics: Moving Vehicles on the Connection Machine." *Proceedings of the 1994 ACM/IEEE Conference on Supercomputing*, November 14-19, 1994.
- [11] Wang, Yibing, Markos Papageorgiou, Albert Messmer. "RENAISSANCE: A Real-Time Freeway Network Traffic Surveillance Tool." *IEEE 9th International Intelligent Transportation Systems Conference*, September 17-20, 2006.
- [12] Redmill, Keith A., Umit Ozguner. "VATSIM: A Vehicle and Traffic Simulator." *IEEE 2nd International Intelligent Transportation Systems Conference*, October 5-8, 2001.
- [13] Fellendorf, Martin. "VISSIM: A Microscopic Simulation Tool to Evaluate Actuated Signal Control Including Bus Priority." *64th Institute of Transportation Engineers Annual Meeting*, October 1994.
- [14] MITSIMLab Open Source License. <http://mit.edu/its/MITSIMLabOSnew.html#license>.
- [15] Maroto, Joaquin, Eduardo Delso, Jesus Felez, Jose Ma. Cabanellas. "Real-Time Traffic Simulation With a Microscopic Model." *IEEE Transactions on Intelligent Transportation Systems*. Volume 7, Number 4, December 2006.
- [16] Zhanfeng, Jia, Chao Chen, Ben Coifman, Pravin Varaiya. "The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors." *IEEE 4th International Intelligent Transportation Systems Conference*, February 12, 2001.
- [17] Petty, Karl F., Peter Bickel, Jiming Jiang, Michael Ostland, John Rice, Ya'acov Ritov, Frederic Schoenberg. "Accurate Estimation of Travel Times from Single-Loop Detectors." *Transportation Research, Part 1 (Policy and Practice)*, Volume 32A, #1, January 1998.
- [18] Vlahogianni, Eleni, Matthew Karlaftis, John Golias, Nikolaos Kourbelis. "Pattern-Based Short-Term Urban Traffic Predictor." *IEEE 9th International Intelligent Transportation Systems Conference*, September 17-20, 2006.
- [19] Quek, Chai, Michel Pasquier, Bernard Boon Seng Lim. "POP-TRAFFIC: A Novel Fuzzy Neural Approach to Road Traffic Analysis and Prediction." *IEEE Transactions on Intelligent Transportation Systems*, Volume 7, Number 2, June 2006.
- [20] Zhanfeng, Jia, Pravin Varaiya, Chao Chen, Karl Petty, Alex Skabardonis. "Maximum throughput in LA freeways occurs at 60 mph." *Berkeley Technical Report*, January 16, 2001.