GrooveNet: A Hybrid Simulator for Vehicle-to-Vehicle Networks

(Invited Paper)

Rahul Mangharam* Daniel Weller* Raj Rajkumar* Priyantha Mudalige† Fan Bai†
*Dept. of Electrical & Computer Engineering
Carnegie Mellon University, U.S.A.
{rahulm, dweller, raj}ece.cmu.edu
†General Motors Research & Development
General Motors Corportation, U.S.A
{priyantha.mudalige, fan.bai}gm.com

***Abstract*— Vehicular networks are being developed for efficient broadcast of safety alerts, real-time traffic congestion probing and for distribution of on-road multimedia content. In order to investigate vehicular networking protocols and evaluate the effects of incremental deployment it is essential to have a topology-aware simulation and test-bed infrastructure. While several traffic simulators have been developed under the Intelligent Transport System initiative, their primary motivation has been to model and forecast vehicle traffic flow and congestion from a queuing perspective. GrooveNet is a hybrid simulator which enables communication between simulated vehicles, real vehicles and between real and simulated vehicles. By modeling inter-vehicular communication within a real street map-based topography it facilitates protocol design and also in-vehicle deployment. GrooveNet's modular architecture incorporates mobility, trip and message broadcast models over a variety of link and physical layer communication models. It is easy to run simulations of thousands of vehicles in any US city and to add new models for networking, security, applications and vehicle interaction. GrooveNet supports multiple network interfaces, GPS and events triggered from the vehicle's on-board computer. Through simulation, we are able to study the message latency, and coverage under various traffic conditions. On-road tests over 400 miles lend insight to required market penetration.**

I. Introduction

Vehicle-to-vehicle multi-hop wireless communication holds the promise of making the driving experience safer, more efficient and more enjoyable. By locally broadcasting safety messages, a disabled vehicle is able to significantly reduce the time to alert all oncoming vehicles. One goal of the inter-vehicle network is to alert and inform vehicles of on-road incidents so they may avert danger and avoid delays by selecting alternate routes. The key performance metric for safety messaging is the time it takes to deliver the message to vehicles in the vicinity and en route to the incident. A secondary goal is to provide support for on-line traffic and road-condition monitoring. By passively exchanging speeds experienced on different road segments, vehicles are able to probe the current degree of congestion along various routes and estimate the time of arrival at the destination along the fastest route. The metric of interest here is the maximum distance of message coverage so travel time estimates may be statistically sound. Finally, there is growing interest in delivering commercial multimedia applications for download via peer-to-peer or infrastructure-based network architectures. The time to setup and maintain a stable connection across all vehicles along the path is of primary concern here. The research challenge is to identify the constraints and degrees of freedom unique to such topology-bound networks and to design vehicle-to-vehicle protocols that perform well across a large range of vehicle densities.

For the above three application categories, it is essential to evaluate the performance and scalability of vehicle-to-vehicle networking protocols in large cities, suburban and rural topologies. In addition, it is crucial to investigate the effect of incremental deployment of such technologies on the message delay, coverage and persistence in the region of interest. As it is expensive to develop and test experimental protocols on a large fleet of vehicles, there is a need for a vehicular network simulator which faithfully models the first-order effects of the street topology, vehicle congestion, speed limits, communication channels and spatio-temporal trends in traffic intensity on the performance and reliability of vehicle-to-vehicle networking. As protocols are designed and evaluated to be suitable via simulation, it is necessary to measure their performance with real vehicles and realistic traffic densities. While it may be possible to deploy a small fleet of vehicles (e.g. a dozen), it is currently not possible to assess the scalability of such protocols in rush-hour bumper-to-bumper vehicle densities.

In order to address the above needs, we present GrooveNet, a hybrid vehicle-to-vehicle network simulator which is capable of communication between simulated vehicles, real vehicles and between real and simulated vehicles. GrooveNet enables us to evaluate the same implementation of vehicular network protocols in simulations consisting of thousands of virtual vehicles on a street map topology and between a test-bed of real vehicles equipped with global positioning systems (GPS) and wireless network interfaces. In addition, GrooveNet supports communication between real and simulated vehicles such that vehicles in the vicinity of each other are able to exchange packets. This approach to simulator design is tailored for rapid development, correctness and stress testing of vehicular network protocols and prototyping of test-beds for multi-hop communication on the road. GrooveNet is also designed to further explore the use of the standards-based Dedicated Short Range Communication (DSRC) [1] and supports simulations across multiple-channels and Denso-based DSRC network interfaces.

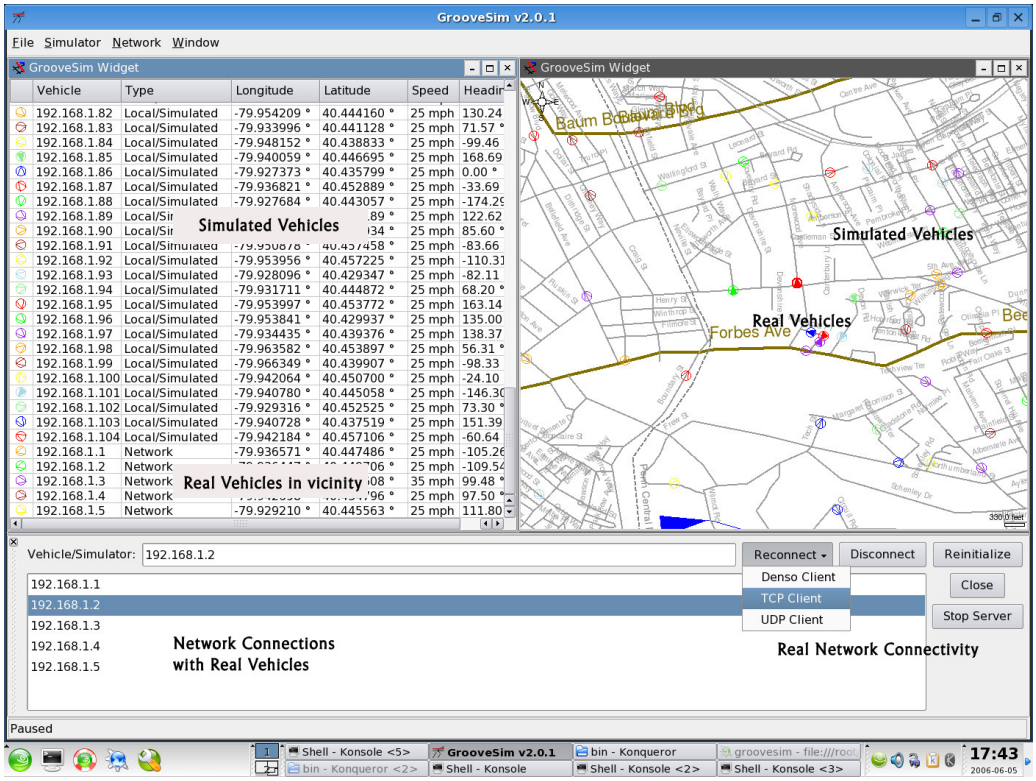

Fig. 1. Real and simulated vehicles interact in the GrooveNet hybrid simulator

GrooveNet is a street-map based vehicular network simulator with the following features:

1) GrooveNet is a modular event-based simulator with well-defined model interfaces that make adding models easy. Models may be added without concern of conflicts with existing models as dependencies are resolved automatically.
2) GrooveNet supports multiple vehicle, trip and mobility models over a variety of network link and physical layer models. In order to correctly represent vehicle interaction, GrooveNet includes simple car-following, traffic lights, lane changing and simulated GPS models.
3) The graphical interface makes it easy to auto-generate simulations consisting of thousands of vehicles across any location in the US. All vehicles obey street speed limits and are displayed on the street map based on their current GPS coordinates.
4) GrooveNet supports three types of simulated nodes: vehicles which are capable of multi-hopping data over one or more DSRC channels, fixed infrastructure nodes and mobile gateways capable of vehicle-to-vehicle and vehicle-to-infrastructure communication.
5) GrooveNet supports multiple message types such as GPS messages, which are broadcast periodically to inform neighbors of a vehicle's current position, and vehicle emergency and warning event messages with priorities. Multiple rebroadcast policies have been implemented to investigate the broadcast storm problem.
6) GrooveNet supports multiple network interfaces for real vehicle-to-vehicle and vehicle-to-infrastructure communication such as: a 5.9GHz DSRC interface, IEEE 802.11a/b/g, 1xRTT and EVDO cellular interfaces. Communication may be established over TCP or UDP sockets. All real vehicles communicate with DSRC or 802.11 with each other and in addition, mobile gateways communicate with infrastructure nodes over the cellular interface.
7) GrooveNet is able to support hybrid (i.e. communication between simulated vehicles and real vehicles on the road) simulations where simulated vehicle position, direction and messages are broadcast over the cellular interface from one or more infrastructure nodes. Real vehicles communicate with only those simulated vehicles which are within its transmission range.
8) GrooveNet is able to connect to the vehicle's on-board computer and read OBD-II diagnostic codes. Events such as sudden deceleration, braking, air bag deployment and signals from the anti-lock braking system can trigger alert or warning messages.

Fig.1 presents a screen shot of GrooveNet implemented in Linux. In the top left panel we observe the list of simulated and real vehicles with their current position, street speed and heading. The top right panel provides a visualization of the current position and heading of vehicles in the city of Pittsburgh, PA, USA. Vehicles marked in solid color have an alert message and are actively rebroadcasting the message. The bottom panel provides network connectivity via Denso-based DSRC interface, UDP communication over 802.11 and TCP connections over cellular interface. During this test we drove five real vehicles around Forbes Avenue in Pittsburgh, USA.

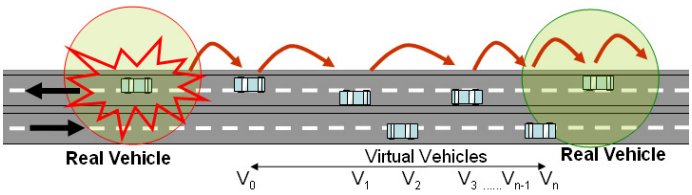

Fig. 2. A hybrid simulation with two real vehicles communicating across multiple simulated vehicles

In Fig.2 we observe two real vehicles which are not within direct communication range of each other. The leading vehicle on the left has experienced a traffic incident (e.g. air bag deployment) which triggers it to broadcast an alert event message. The message is forwarded to the DSRC and cellular interfaces. As no vehicle is within range, the message over the DSRC interface is not received by any real vehicles. The message is also forwarded over the cellular interface to a remote server. The remote server, also known as the Vehicle Operations Director (VOD), simulates all simulated vehicles in the vicinity and updates the network state with the triggered event. Over time, the simulated vehicles in range of the incident vehicle propagate the message hop-by-hop over their virtual DSRC interfaces. When a simulated vehicle receives the message and is within the default communication range of a real vehicle, the remote server forwards the event message to the real vehicle's IP address. This way, an event triggered by one real vehicle is received by another via multi-hop communication across one or more simulated vehicles in the same geographic vicinity. All vehicles follow the same rebroadcast policy, street speed limit and obey car-following. The vehicle density can be increased arbitrarily and its effects can be observed by a driver in a real vehicle on the road. Only a subset of real vehicles need operate as mobile gateways.

The ability to communicate between a small number of real vehicles and a large number of simulated vehicles using the same protocol implementation, algorithms and packet types provides several benefits. It allows for rapid prototyping and evaluation with a real wireless channel, transmission ranges, link layer arbitration and network stack delays. Hybrid simulation provides application users with an intuitive feel of the impact of communicating vehicle density on packet delivery ratio and event response time. Furthermore, it provides the developer with feedback on the accuracy and the details necessary in the simulation models.

A. Organization of this Paper

This paper is organized as follows: in Section II we present related work followed by an architectural overview of the simulator and test-bed design in Section III. In Section IV we detail the models incorporated in GrooveNet. Sections V presents the simulator's performance and availability followed by the conclusion.

II. Related Work

Vehicular networks are a special case of mobile ad hoc networks (MANET) where the topology is constrained to a street map, maximum relative speeds are in excess of 90m/s and the node density spans a large dynamic range reaching over 5,000 vehicles/mi^2 [2]. Over the past two decades under the auspices of MANET, there have been a large number of path-based end-to-end routing protocols [3], [4]. Most protocols have been evaluated using arbitrary and unrealistic mobility models such as random walk, random walk with reflection, random walk with wrapping, random waypoint and probabilistic versions of random walks with correlated speed and direction. Several mobility studies [7]–[9] show that the results obtained from different mobility models vary widely and do not realistically represent vehicular traffic speeds, directions and trips. In [5], an ns-2 simulation comparison of random waypoint and mobility in topology-constrained urban concludes that random waypoint is a good approximation for vehicular networks. They, however, choose a dense network with an average degree of 20 nodes and an average path length of just 3 hops and do not consider car-following or traffic control through traffic lights or stop signs. To improve performance of routing protocols they increase the radio transmission range to $1/2$Km. The authors in [6] implement basic car following and probabilistic traffic control and compare their effect on routing performance to that of random waypoint.

Most MANET protocols specify path-based routing mechanisms with an aim to establish a connection between a source node and one or more destination nodes [3]. In [19], such protocols are demonstrated to be unsuitable for vehicular networking because they do not provide stable paths across multiple vehicles for even moderate durations. At the relative speeds experienced in vehicular networks and the fast rate of topology change, it is important to minimize any handshaking and shared state among nodes. Ad hoc routing protocols such as DSR [4] have been shown to perform poorly [7] at node speeds above 20m/s and experience packet delivery ratios of less than 50%. Furthermore, in the vehicular networking context, messages are targeted to vehicles based on their position, heading and speed and hence broadcast protocols are more suitable than point-to-point routing.

Several simulators such as ns-2 [11], GloMoSim [12] and QualNet [14] have been developed for generic ad hoc networks and model the wireless channel with increasing detail. They however do not support vehicle network specific topologies and traffic control models. CORSIM [16] and PARAMICS [18] are microscopic traffic simulators that provide highly accurate traffic queuing and vehicle interaction models. They focus on transportation planning and road network design and do not integrate any vehicle-to-vehicle communication. Several studies [15], [17] have coupled a simulator from the first and the second category to evaluate the performance of vehicular network protocols and have expressed the need for a scalable simulator specialized for vehicular communication. QualNet, CORSIM and PARAMICS are proprietary simulators and require purchase of a software license. To the best of our knowledge, none of the above simulators implement interfaces to interact with real nodes which use the same protocol

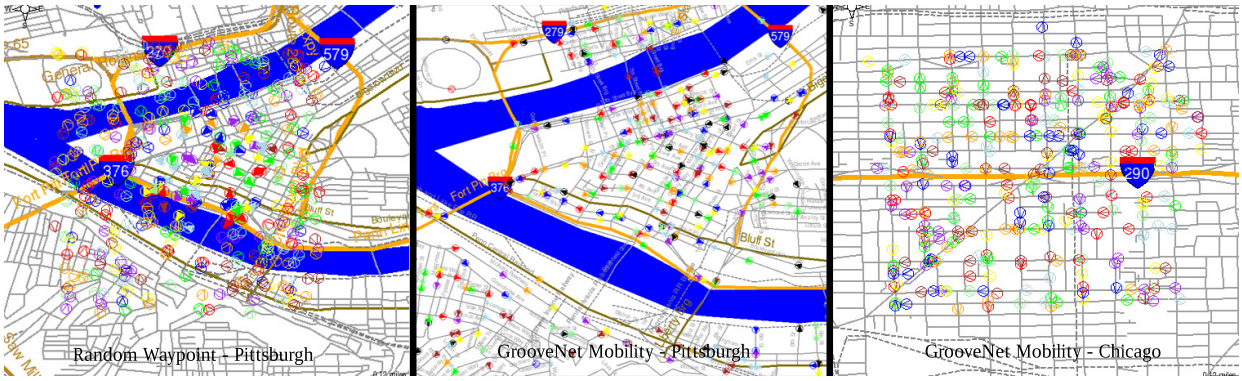

Fig. 3. Comparison of Random Waypoint mobility model with GrooveNet mobility model in Pittsburgh and Chicago

implementations.

A. Impact of Topology on V2V Performance

In order to better understand the effect of a constrained topology, such as a street map, and the impact of traffic control mechanisms, such as car-following and traffic lights, we compare the message propagation rate with GrooveNet to that of the random waypoint mobility model. We looked at the difference in performance across a range of street and vehicle densities and focus on dense urban areas of Chicago and Pittsburgh with a vehicle density of 100 vehicles/km^2 and rural environments with a lower density of 20 vehicles/km^2. The random waypoint model was implemented in the simulator as an unconstrained mobility model where vehicles chose a random direction, traveled at a speed s for a duration d and paused for a duration p before choosing another random direction to traverse. We choose s to be uniformly distributed between 25miles/hr and 35miles/hr, d=20 seconds and p=1 second. On the other hand, vehicles using the GrooveNet model employed the car-following model and were not permitted to go over each other. They used the random walk model in GrooveNet and were constrained to drive on the streets at the speed limit which was between 25 and 35miles/hr. Initially, all vehicles were distributed randomly and a message was broadcast from a vehicle close to the center. The message propagation ratio is the ratio of the number of vehicles which have received the message via broadcast from vehicle-to-vehicle communication to the total number of vehicles in the region of interest. We repeated the experiment 5 times over an area of 4km^2 and observed the variation in message propagation to be less than 10% across all runs. In order to isolate effects of other models we fixed the transmission range to 200m and considered the wireless channel to be error-free and collision-free. Vehicles which have received the message forwad it once every second.

In Fig.3 we observe vehicles in urban areas using the random waypoint model, the GrooveNet car-following and traffic light models in Pittsburgh and Chicago respectively. Vehicles using the random waypoint model were free to traverse the area without concern for streets or water bodies. In Fig.4 we observe the random waypoint model and GrooveNet mobility models in a rural area. Fig.5 shows the message propagation as a function of time. We note that in both urban and rural cases, the random waypoint model results in optimistic behavior with a faster rate of message propagation. On the other hand, the street model enforces disconnects in the vehicle network graph and results in a slower rate of propagation. The area of interest in Chicago closely approximates a grid of streets with a uniform vehicle distribution and hence performs similar to the unconstrained model. On the other hand, the message propagation is slower in Pittsburgh where the streets are more irregularly spaced and water bodies are present.

In the rural experiments, we observe that vehicles with the GrooveNet mobility model occupy only a fraction of the area and are hence more rarefied. The message propagation is much slower than the unconstrained case as vehicles are less

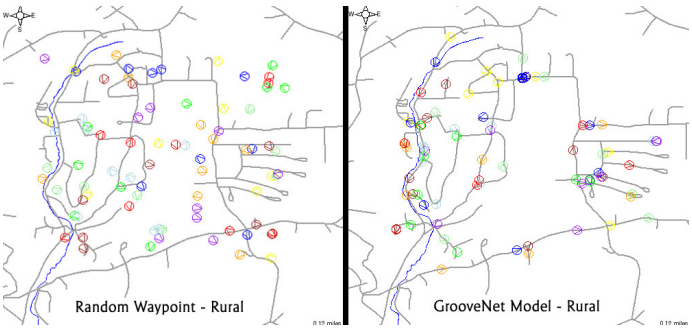

Fig. 4. Comparison of Random Waypoint and GrooveNet mobility models in rural Allegheny county, Pennsylvania. USA.

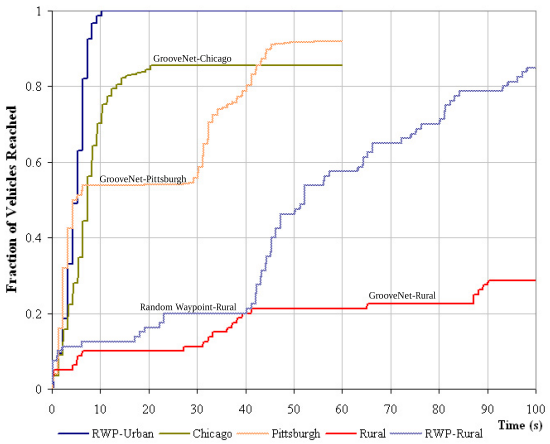

Fig. 5. Message penetration with Random Waypoint and GrooveNet mobility models in urban and rural areas

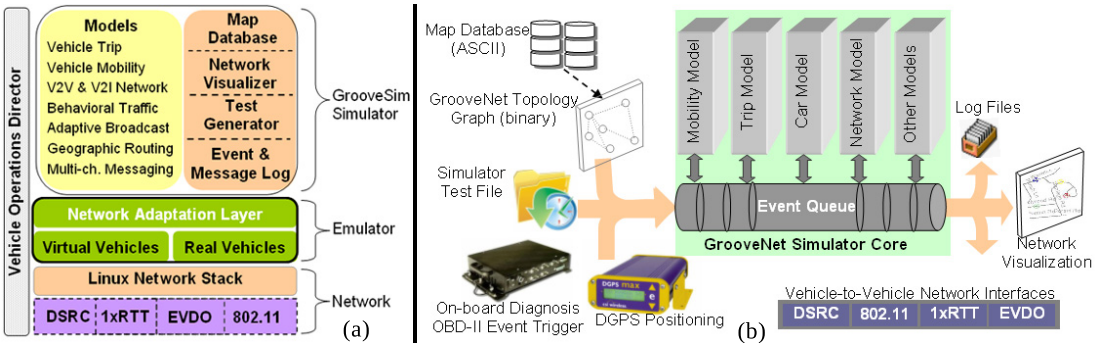

Fig. 6. GrooveNet Hybrid Simulator engine with input and output components

uniformly distributed. This results in a largely disconnected network and clearly highlights the difference between the mobility models. We also observe intervals where the number of informed vehicles stalls due to the separation between streets. The slower message propagation can be explained by observing the distribution of number of neighbors within transmission range for each mobility model. For the random waypoint model in urban areas, the distribution of neighbors had a narrower spread between 9-12 and a mean of 10 vehicles while the GrooveNet model resulted in a less uniform distribution with several large clusters of 22-26 neighbors. In the rural case, the random waypoint model had a smaller mean between 3 and 4 neighbors while the GrooveNet model resulted in larger clusters of 5-6 neighbors. The random waypoint model's deviation from the topology-constrained model is more pronounced at lower vehicle densities and when the topology is composed of several strong components.

III. Hybrid Emulator Architecture

GrooveNet has been designed to function both as a simulator and as a test-bed for on-road experiments. In order make the simulator portable to a test platform, we use the same implementation of networking policies and frame types in both modes. As shown in Fig. 6(b), there are five primary inputs into the hybrid simulator engine. The map database and simulation scenario test file are required for the simulation mode. In addition, the GPS receiver, one or more network interfaces and an optional on-board diagnosis sub-system are necessary for the the test-bed. The hybrid simulator is able to output data in three forms: the current location of all vehicles is displayed on a street map visualization; all communication, events and vehicle data can be written to one or more log files; and packets can select and transmit over one or more interfaces for real vehicle interaction. The simulator core is composed of a model manager with which models register and an event queue which schedules events for each vehicle and its models. We now describe the simulator mode followed by the test-bed and hybrid mode.

A. GrooveNet Simulator Mode

The simulator mode provides models that approximate vehicular communication and mobility to the first order and make it easy to add new custom models. GrooveNet is composed of four major components as shown in Fig. 6(a): The simulator, vehicle network emulator, network and device interfaces and the VOD. The simulator is composed of multiple vehicle, communication, environment, routing and rebroadcast models and is detailed in Section IV. The simulator can operate without initializing the other components and uses a subset of the models. As shown in Fig. 7, large scale simulations with thousands of vehicles can be auto-generated by selecting an area with the mouse in any region of the map and specifying which models to associate with the vehicles. The auto-generator will then create a simulator test file and list all the vehicles, their start and end points, and their associated models and parameters. When the test file is loaded, the simulator resolves all dependencies between models, creates all vehicles and initializes them in the event queue.

Vehicles may be placed at specific or random addresses on the map. All updates of the current status of each vehicle and each packet are logged in multiple log files for easy graph plotting. It is easy to add new log files and new items in a log file through built-in functions. The current GPS coordinates, speed, direction of all vehicles are displayed as a list and on the map. We now describe the modular architecture and network abstraction layer which facilitates communication with real and simulated vehicles.

1) Modular Architecture: GrooveNet is supports multiple models and is designed to be extensible so custom models for security, applications, broadcast and routing protocols, etc. may be easily added. The Model Manager maintains a registry of all loaded models and is shown in Fig. 8. It contains a list of model creation functions for each model type, a list

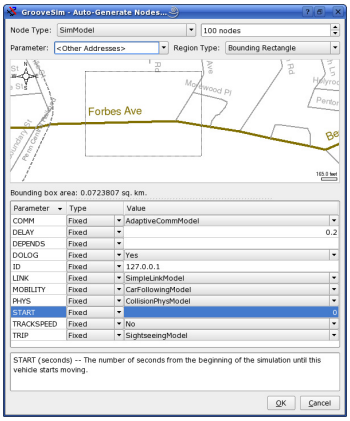

Fig. 7. Vehicle and model auto-generation in the specified map region

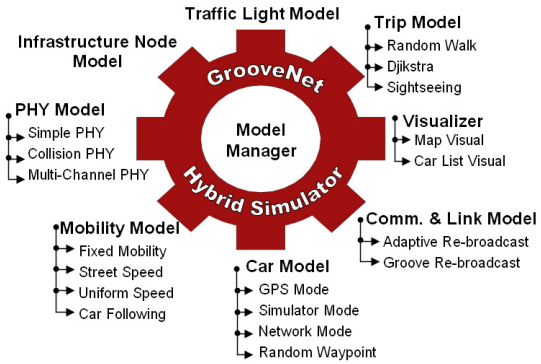

Fig. 8. GrooveNet Model Manager with inherited models

```
int CarFollowingModel::Init(map<QString, QString> & mapParams)
{
  // Called on startup. Adds model to the simulator event queue.
}

int CarFollowingModel::PreRun()
{
     // Called at the beginning of each simulation run. Reset internal state.
}

int CarFollowingModel::ProcessEvent(SimEvent & event)
{
     // Called when an event is dispatched by the simulator event queue
}

int CarFollowingModel::PostRun()
{
     // Update log files, visualization and send packets to net interfaces
}

int CarFollowingModel::Cleanup()
{
     // Free memory and remove model from registry and event queue
}
```

Fig. 9. Model life-cycle with simple interface to add models

of pointers to each existing model instance and each model's dependencies on other models. During initialization, the Model Manager resolves all model dependencies by constructing a model dependency tree and initializes each model only after its parent model has been initialized. All models are derived from one or more abstract model classes and define virtual functions. GrooveNet currently defines eight abstract model classes and several derived models with different implementations. By extensive use of C++ polymorphism, a new model just needs to extend one or more abstract model types and implement the specific behavior.

2) Model Life-cycle: We briefly describe the process of adding a new model to GrooveNet. The model needs to be registered with the Model Manager by specifying a pointer to its model creator function and model name. As illustrated in Fig. 9 each model must implement five functions for initialization, pre-run, main event process, post-run and clean-up. This simple interface makes it easy to derive and add new model implementations without the need to understand the inner workings of the simulator engine.

3) Network Abstraction Layer: All vehicles periodically locally broadcast HELLO messages with their coordinates. In addition vehicles exchange traffic incident messages and congestion probe messages. The user can specify the message type, priority, the geographic region within which it may be rebroadcast and the lifetime. Messages may be sent on one shared channel or individual channels as specified by the DSRC standard.

The Network Abstraction Layer (NAL) is an interface that allows the simulator to treat the network as an abstract entity, whether the network exists only on one machine or multiple. The NAL exposes two functions to the simulator: TransmitPacket() and ReceivePacket() which communicate with both local and remote vehicles using the same packet format. Message exchanges over the network are handled by generic Client and Server classes from which TCP and UDP servers are inherited. Thus TransmitPacket() and ReceivePacket() call Write() and Read() in the inherited Client/Server class and the messages are forwarded to the instantiated interface. The messages are forwarded to all local simulated vehicles and through multiple inheritance packets are forwarded to any active interface. GrooveNet currently supports vehicle-to-vehicle communication over an 802.11a interface that has been modified to suit the 10MHz channel bandwidth and 5.9GHz center frequency of the DSRC specification. 802.11a/b/g is also supported for vehicle-to-vehicle communication. In order to communicate with infrastructure nodes or to the VOD, GrooveNet supports 1xRTT and EVDO cellular modems. It is easy to add a new interface by extending the existing Client/Server class.

4) Network Visualization: Simulations are much more exciting if the user can see the network topology and the connectivity between vehicles. GrooveNet lists all vehicles and infrastructure nodes with their GPS location, direction and speed (Fig. 1). In addition, all nodes are displayed on a map with streets, street names and water bodies. Vehicles are represented as hollow triangles. When a vehicle receives a traffic incident message it is displayed as a solid triangle. With this visualization, the user is able to see the spatial propagation of the message. The graphical user interface allows the user to initialize TCP/UDP servers and connect to real vehicles.

B. GrooveNet On-Road Test-bed & Hybrid Mode

Our test-bed consists of 5 vehicles, each with a Linux-based laptop running GrooveNet, a Denso-based DSRC interface with a magnetic mount antenna, a Verizon EVDO cellular interface, a CSI-Wireless Differential GPS receiver and headsets for voice communication between vehicles. In hybrid mode, a remote desktop was placed in Warren, Michigan with our team at General Motors while the vehicles traveled 300 miles away in Pittsburgh. Both real vehicles and simulated vehicles running on the remote machine were displayed on the same screen. When a simulated vehicle was in the transmission range of a real vehicle, the real vehicle displayed messages received from the simulated vehicle. Our goal was to reliably create hybrid simulations with over 200 vehicles with 1Hz

Fig. 10. The authors with the GrooveNet on-road test-bed

update rate as the cellular link bandwidth was limited to an average of 400Kbps.

IV. Vehicular Network Models

We describe traffic, communication and topology models as show in Fig. 8. Our goal was to capture the basic models and to facilitate addition of more realistic models over time.

A. Graphical Street Map Topology

GrooveNet is based on the US Census Bureau's TIGER/Line 2000+ database format and is able to dynamically load counties at run-time. On startup GrooveNet reads map database text files and converts the topology data into a binary encoded file with a graph structure. This approach of using an intermediate format provides three benefits: (a) by constructing a non-negative weighted edge graph with road, rail and water segments as edges and street intersections as vertices, we are able to execute vehicle routing and path planning algorithms with ease, (b) the binary format speeds up the traversal and display of the topology and (c) the intermediate format is independent of the map database format and hence allows GrooveNet to be easily ported to maps of Europe and Japan. Traffic lights are embedded at intersections of non-highway roads within the map database and have a parameterized duty-cycle.

B. Vehicle, Mobile Gateway and Infrastructure Models

Several vehicle types are used in GrooveNet: the infrastructure node, a simulated vehicle with vehicle-to-vehicle communication, a node gateway node with additional vehicle-to-infrastructure communication and a simulated vehicle not linked to the map database (unconstrained). The infrastructure nodes are assumed to be linked with each other via a wired network and share a common state. Infrastructure nodes are able to receive packets only from gateway nodes but broadcast data to all communication-enabled vehicles. Simulated vehicles are described by their origin, destination, mobility model, trip model, communication range and link layer.

C. Communication Models

GrooveNet supports multiple physical and link layer models. A threshold-based physical layer with a uniform error rate is the base physical layer model. It is extensible to channel models we derived with our on-road experiments with fast fading and log-normal path loss. Packet collisions and filtering based on geographic location are modeled in the link layer.

D. Adaptive Rebroadcast Models

The link layer supports several adaptive rebroadcast models to alleviate the broadcast storm problem [22]. When a traffic incident message is broadcast from the source vehicle, each scheme throttles the rebroadcast rate based on its position, distance from the event, number of active neighbors, etc. The First Fast Rebroadcast scheme forwards the packet the first time with a small delay (e.g. 500us) and subsequent rebroadcasts have a longer interval (e.g. 1 second). This results in a transient broadcast storm but initially propagates the message quickly. The distance-based rebroadcast rate control selects the rebroadcast rate as a function of the current vehicle's distance from the event location. The rebroadcast rate is high near the event (e.g. 500ms) as it is most relevant to alert drivers near the incident. The rate decreases linearly or exponentially with distance to a maximum interval (e.g. 5 sec) determined by the maximum relative vehicle speed. With the position-based rebroadcast suppression a vehicle does not transmit during an interval if it overhears a broadcast from a vehicle further away from the event. This is relevant because the message has already propagated down the road beyond the vehicle's current location. Finally, the neighbor-based duty cycle throttling scheme increases the rebroadcast interval exponentially based on the number of rebroadcasting neighbors the vehicle overhears. We used the binary exponential backoff with a minimum window of 500us. [22] shows that additional spatial coverage is only 19% when message is heard from 2 neighbors and $<$0.05% when message is heard from $>$4 neighbors.

TABLE I
Link Utilization of Adaptive Rebroadcast Schemes

Distance	Periodic	First-fast	Location	Distance	Neighbor
0.5km	0.666	0.462	1.906	1.30	0.559
1km	2.03	0.466	2.76	3.58	3.26
1.5km	2.92	0.468	3.93	4.22	3.81
Rate	4.39	4.6	0.67	2.67	0.31

We evaluated the above four rebroadcast schemes by observing the trade-off between message delay and the rebroadcast duty cycle function (link utilization). We choose an urban area in Pittsburgh with a vehicle density of 25 vehicle/km^2. The link layer modeled packet collisions. A message was broadcast from an event vehicle and the delay and message receive rate (messages/(vehicle*sec)) was recorded. In Table 1, we observe that the First Fast Rebroadcast scheme has the smallest delay but the highest message receive rate. The neighbor-based rebroadcast scheme provided th best trade-off between flooding the network and end-to-end message delay. The periodic scheme rebroadcast message every 1 second.

E. Mobility Models

GrooveNet supports multiple mobility/speed models for vehicle travel through a street map. In the Car-following model [23], a vehicle will not exceed the speed of a vehicle in front. A vehicle that is determined to be a leader vehicle will use a different mobility model such as Street Speed model. With the Street Speed mobility model vehicles always move within user defined range (e.g. +25%, -25%) of the speed limit of the road. With the the Fixed mobility model, vehicles do no move. This is useful for infrastructure nodes and for tests that do not require dynamic movement.

F. Trip Models

GrooveNet supports the Random Walk model, Djikstra's minimum weight routing and the Sightseeing model with movement constrained to streets. In the random walk model, a vehicle traverses a segment until the next intersection and then randomly decides the a direction and segment to take next.

Vehicles are biased against going back on the same segment. The Djikstra route planning model finds the minimum weight path where the weights are the speed limits of street segments. While the models presented are basic approximations of vehicle movement, additional models can be added using the methods discussed previously. The Sightseeing trip model tries to mimic a driver traveling from her home on a set of errands and then returning home. The vehicle randomly walks until it is a certain distance from the starting point. The vehicle then takes the shortest path back to the starting point and starts again along a different path.

V. Simulator Performance and Distribution

GrooveNet is implemented in C++ and Qt [24] graphics cross-platform library in Linux. To evaluate the scalability of the simulator, we timed the real-time taken by a 1.8GHz Pentium M with 1GB RAM to simulate 10 seconds of simulation for different number of vehicles. We choose the most interactive models (car following, packet collisions, sightseeing trip model) and distributed vehicles in a small area of 1Km^2. This ensured that all vehicles interact with each other and the worst case performance was determined. Fig. 11 shows the real-time it takes to execute simulations from 10 to 4000 vehicles. Given the map database lookups, visualization refreshes and complex interaction between models, the simulator scales well to networks with of thousands of vehicles.

All models have been validated for correctness in a suite of validation tests. Furthermore, on-road driving experiences with real and simulated vehicles verified the speed, transmission range and delay along one and more hops. In order to run statistically sound tests, GrooveNet supports a Monte Carlo multiple-run test feature where vehicle movement varies from test to test and events may be scheduled a priori.

VI. Conclusion

In order to demonstrate the viability of vehicular networks and investigate new multi-hop protocols, we present GrooveNet - a hybrid simulator for vehicular networks. GrooveNet is composed of several models that characterize communication, travel and traffic control to enable large scale simulations in street maps of any US city. GrooveNet supports interaction and communication between real and simulated vehicles. This approach to vehicular network platform design provides the following benefits:

- It helps evaluate the necessary vehicle density for target end-to-end message delay, propagation and persistence in the region of interest.

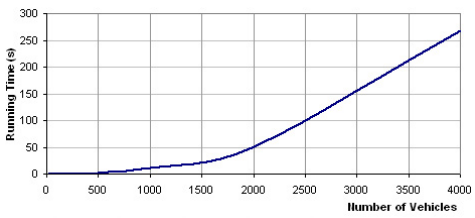

Fig. 11. Simulator performance

- As the same models, rebroadcast algorithms and packet formats are used in simulation and in the test-bed, GrooveNet enables rapid protocol design and evaluation with real world traffic patterns and wireless channels.
- The extensible modular design allows for easy addition of custom models including those for networked applications, security, link and routing protocols.

The current limitations are that map database does not indicate one-way streets and the altitude of the street. As future work, we aim to design more realistic models and further validate the faithfulness of the simulator to on-road driving. GrooveNet is available to the research community to help further explore protocol and system design of vehicle-to-vehicle networks.

References

[1] IEEE Standard Specification for Telecommunications and Information Exchange Between Roadside and Vehicle Systems - 5 GHz Band Dedicated Short Range Communications MAC and PHY Specifications.

[2] Mercer County Vehicle Density, 2005. $http://gis.tcnj.edu/maps/mercer/mercer_vehicles.pdf$

[3] E. Royer and C.-K. Toh. A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks. *IEEE Personal Communications Magazine*, vol. 6, no. 2, pp. 46-55, April 1999.

[4] J. Broch, D. Maltz, D. Johnson, Y.-C. Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. *Proceedings of ACM MobiCom*, pp. 85-97, Oct 1998.

[5] Saha, A. K. and Johnson, D. B. Modeling mobility for vehicular ad-hoc networks, *VANET'04*. Philadelphia, USA, Oct. 2004.

[6] Choffnes, D. R. and Bustamante, F. E. An integrated mobility and traffic model for vehicular wireless networks. *VANET '05*, Germany, Sept 2005.

[7] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing*, 2(5):483–502, 2002.

[8] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. *ACM Infocom '03*, pp. 1312-1321, San Francisco. April 2003.

[9] J. Yoon, M. Liu, B. Noble. Sound Mobility Models. *ACM Mobicom*, San Diego, California, 2003

[10] Boxill S. A. and Yu L. An Evaluation of Traffic Simulation Models for Supporting ITS Development. *Technical Report 167602-1. Texas Southern University*, Oct. 2000.

[11] Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/

[12] L. Bajaj, M. Takai, R. Ahuja, and R. Bagrodia. Simulation of large-scale heterogeneous comm. systems. *In MILCOM*, Nov. 1999.

[13] OPNET Technologies, Inc., http://www.opnet.com/.

[14] QualNet, http://www.scalable-networks.com

[15] Wu, H., et. al . Simulated Vehicle-to-Vehicle Message Propagation on Atlanta's I-75 Corridor. Journal of Trans. Research Board 2005.

[16] CORSIM: Microscopic Traffic Simulation Model. http://www.strategicforecasting.com/

[17] Xu, H., and Barth, M. A transmission-interval and power-level modulation methodology for optimizing inter-vehicle communications. *VANET'04*. Philadelphia, USA, Oct. 2004.

[18] Quadstone, Inc. The PARAMICS Transportation Modeling Suite. http://www.paramicsonline.com/ 2004.

[19] J. Blum, A. Eskandarian and L Hoffman. "Challenges of Intervehicle Ad Hoc Networks." *IEEE Transaction on Intelligent Transportation Systems*, 5(4):347–351. 2004.

[20] TIGER/Line 2005. US Geological Survey (USGS) topographic maps http://www.census.gov/geo/www/tiger

[21] Harri, J., Filali, F., Bonnet, C. A framework for mobility models generation and its application to inter-vehicular networks, *IEEE MobiWac*, Hawaii, USA, June 2005.

[22] S.-Y. Ni. et al., The broadcast storm problem in a mobile ad hoc network. In Proc. of ACM MobiCom, August 1999.

[23] Rothery, R. W. Car following models. In Trac. Flow Theory (1992), Transportation Research Board, 165.

[24] Qt Cross-platform GUI Development, TrollTech Inc.