



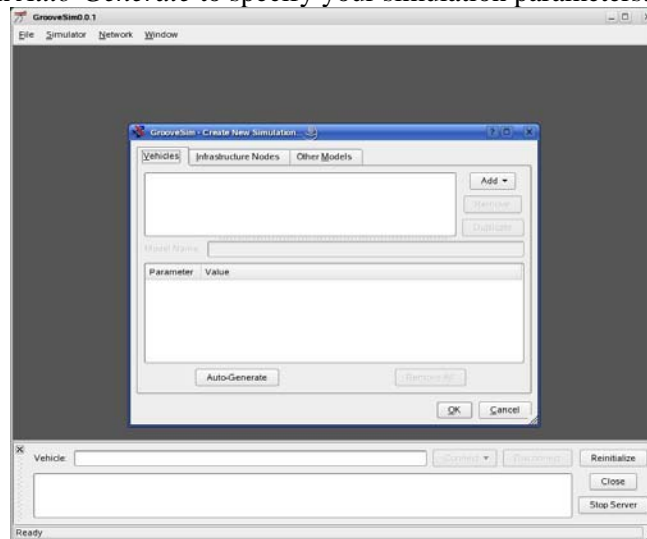
GrooveNet - Simulation Guide

Rahul Mangharam (rahul@cmu.edu)

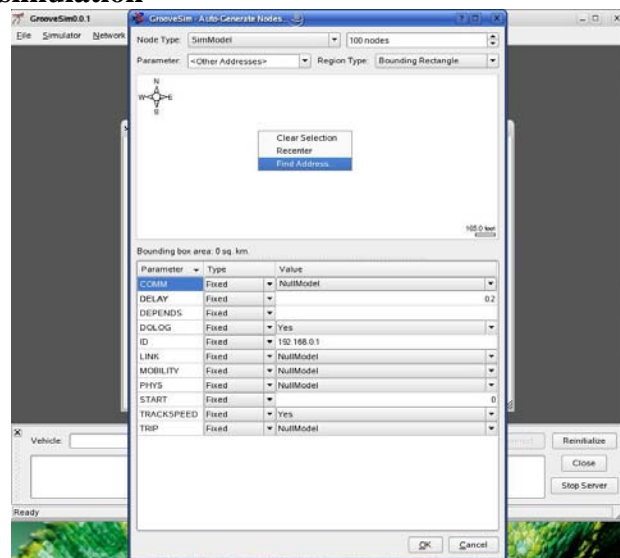
This guide describes the simple steps to install and run a GrooveNet simulation. GrooveNet requires Linux kernel 2.6.* and Qt 3.3 graphics library. SUSE 10 is the preferred Linux distribution.

Step 1: Installation

Follow the steps listed in the **GrooveNet Quick Start Guide** to install GrooveNet. In the groovenet/project/bin directory, type `./groovenet` to run the simulator. The following start-up window appears. Click on *Auto-Generate* to specify your simulation parameters.



Step 2: Setting up the simulation



In the pop-up window, under **Node Type**, select **SimModel** and enter the number of nodes. In the blank bounding box region, right click and select Find Address. Enter the desired address such as *4000 Forbes Ave, Pittsburgh, PA* or *333 7th Ave, New York, NY*. This centers the map to the selected street. Navigate the map using the arrow keys. Use - to zoom out, + to zoom in and = to revert the map.

Step 3: Specify Vehicle Parameters

Use the mouse to select the region where the cars are placed.

Select the parameters for you test. The following models are provided.

Communication Layer

SimpleCommModel
BasicCommModel

Multiple Access Model

SimpleLinkModel

Mobility Model

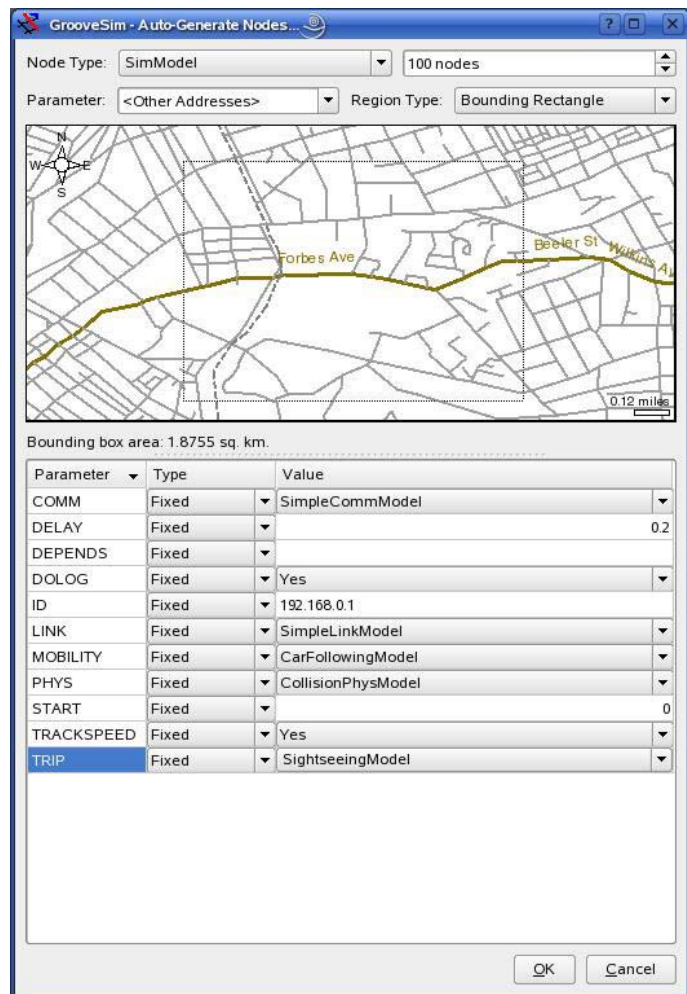
FixedMobilityModel
FixedSpeedModel
UniformSpeedModel
CarFollowingModel

Physical Layer

SimplePhysModel
CollisionPhysModel
MultiPhysModel

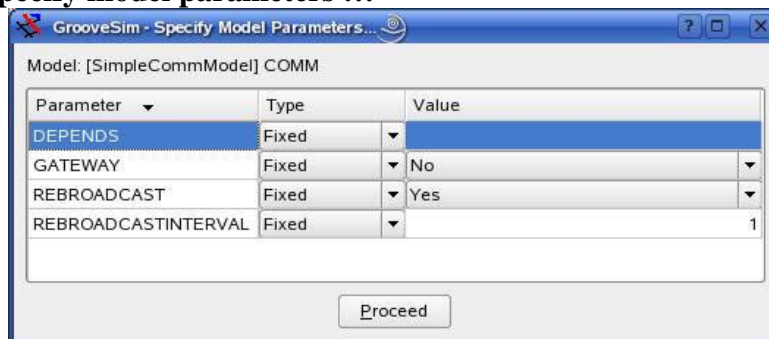
Trip

RandomWalkModel
DijkstraTripModel
SightseeingModel

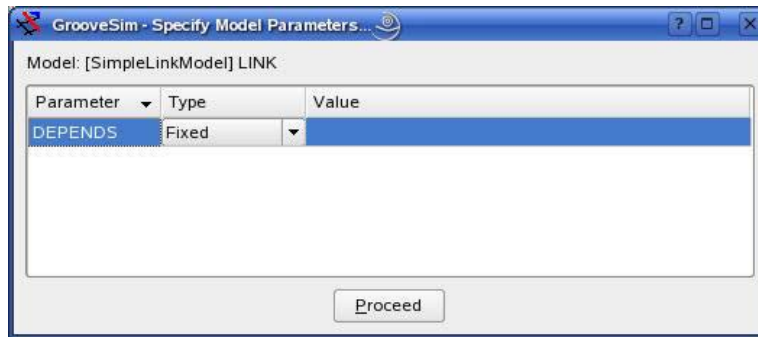


For a simple test, select the parameters as show in the image above. The **SimpleCommModel** accepts all packets and rebroadcasts event messages at a fixed re-broadcast rate. The **CarFollowingModel** implements car following and specifies the characteristics of a “leader” vehicle that blocks all other vehicles on the same street segment behind it. The **CollisionPhysModel** drops packets if concurrent transmissions are heard within the transmission radius of the receiver. The **SightseeingModel** is a trip model where nodes start at a given location and randomly walk until they reach a maximum distance and then find the shortest path back to the start location. They then repeat this process.

Step 4: Click to specify model parameters ...



The re-broadcast interval is listed in seconds.

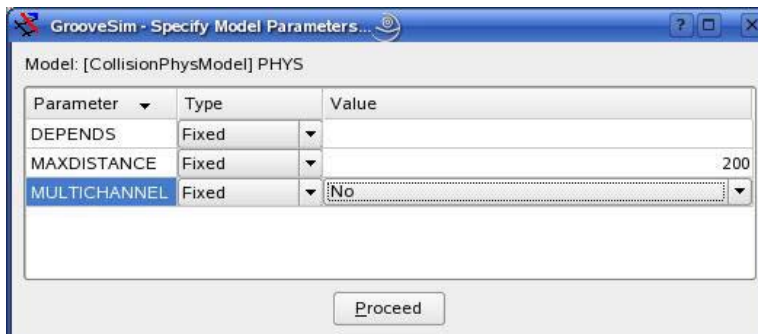


Do nothing here. Click Proceed to go to the next window.

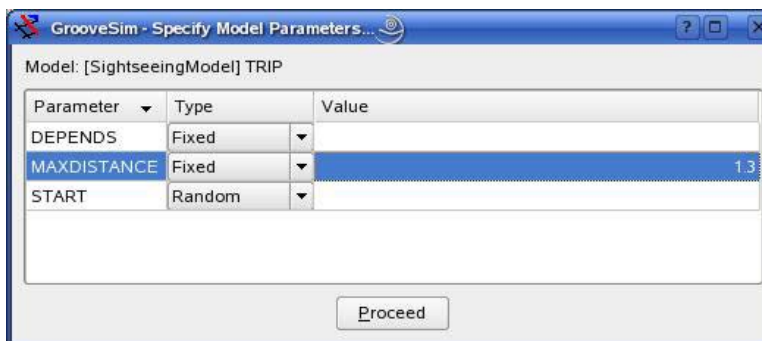
Step 5: Specify the Mobility Model



The car-following model specifies the behavior of the leader vehicle along a street segment. We choose the **UniformSpeedModel** for the leader vehicle to introduce some randomness.

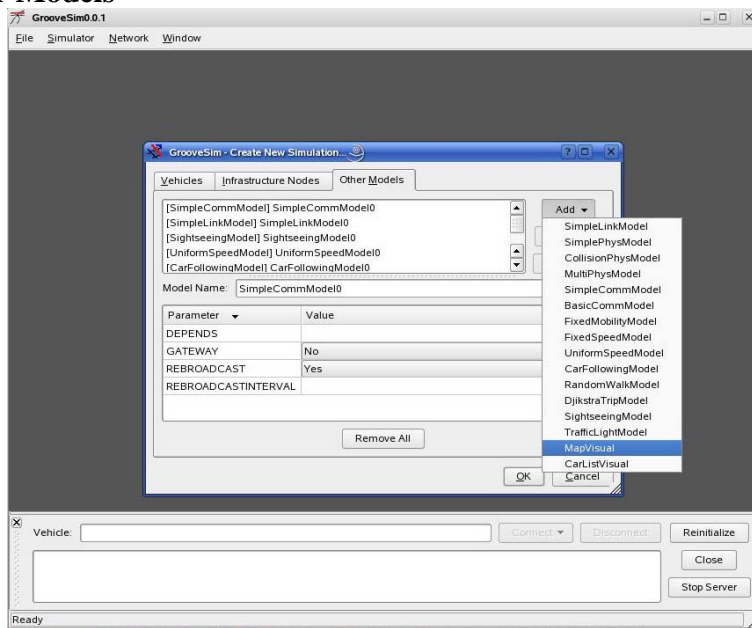


The physical layer model specifies the maximum transmission distance in meters. You can choose to transmit the periodic gps updates and the event message on the same or different channels. In this case, we choose to send both packet types on the same channel.



The **SightSeeingModel** specifies the maximum travel distance from the start position a vehicle may travel before returning. The distance is measured in km and the start location can be specific or random within the bounding box.

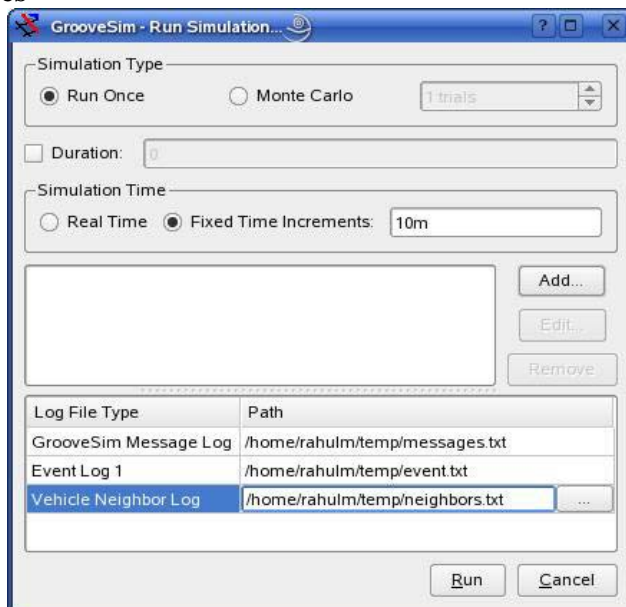
Step 6: Add Other Models



Once all the vehicle model parameters are entered, click on the **Other Models** tab and add the **MapVisual** and **CarListVisual** models. This enables the user to turn on or off the GUI.

Now *Save* the simulation file with a *.sim* extension. To run the simulation, click on Simulator→Run.

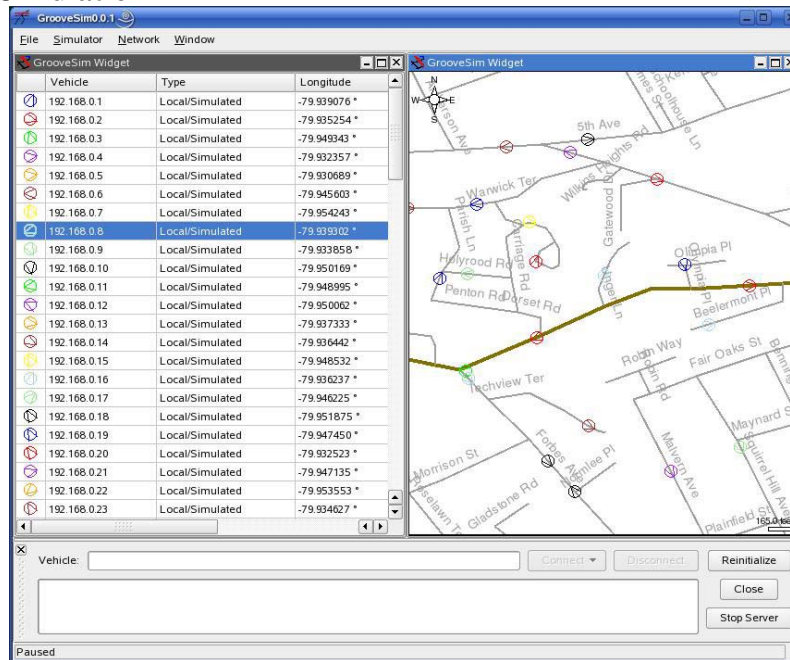
Step 7: Specify Log Files



In the pop-up box, specify the text files to save the message, event and neighbor log files. The *event* log records the progress of the event's messages. The messages log records all the gps messages that have been sent and received by each vehicle during every rebroadcast. Finally, the *neighbor* log records the number of neighbors at each time interval and the number of messages received successfully and number of collisions. This is useful to monitor the link utilization.

It is important to set the **Simulation Time** to *Fixed Time Increments* of 0.7. This is the sim update rate.

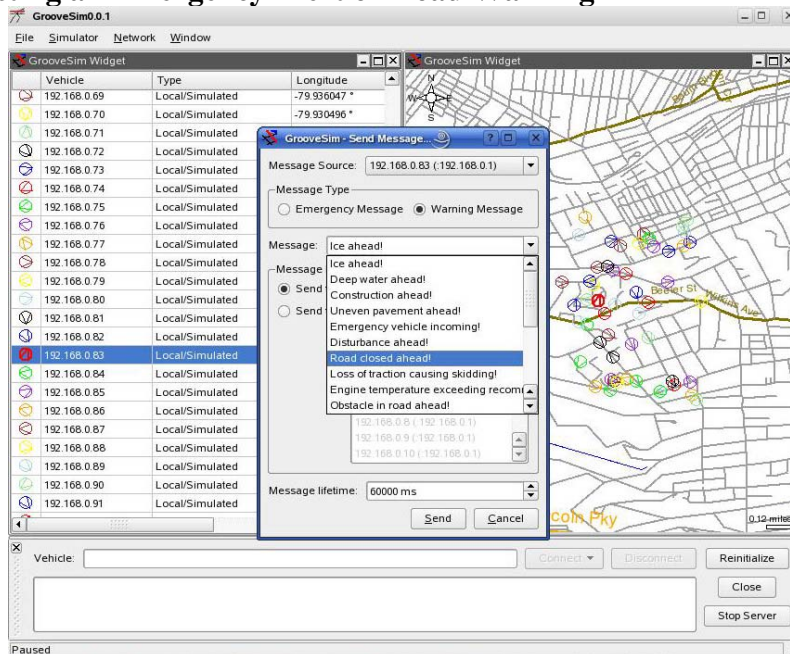
Step 8: Run the Simulation



After the log files are specified, the simulation window will appear. Click on Window→Tile to set the window layout. Then click on a vehicle and drag it to the right **CarVisual** window. You can *pause* and *stop* the simulator from the **Simulator** tab.

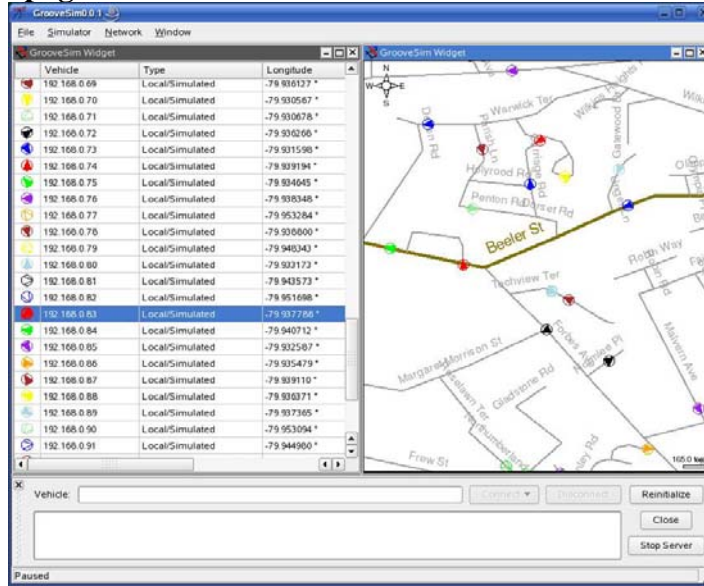
Clicking on a vehicle in the **carList**, re-centers the map on the right to that vehicle. Clicking a vehicle on the map on the right displays its IP address on the bottom left status bar.

Step 9: Broadcasting an Emergency Alert or Road Warning



To send a message from a vehicle, you must right click a vehicle in the car-list. From the pop-up window, you can select the message type, message text and lifetime.

Step 10: Message Propagation



Vehicle that have received the message are displayed in solid colors. You can now see how the message diffuses from the event-origin vehicle.

Using the BasicCommModel

This model uses various message rebroadcast rate adaptation and suppression schemes to combat the broadcast storm problem. The user can specify adaptive rebroadcast schemes (*AdaptiveEnable*) such as the first rebroadcast to have a small jitter. The rebroadcast scheme can be based on distance from the event, the relative location of other re-broadcasters and the number of overheard messages from unique neighbors.

We highlight here the use of the **BasicCommModel** in the geographic region surrounding the GM Technical Center in Warren, MI.

