

# A Three-Tier Communication and Control Structure for the Distributed Simulation of an Automated Highway System \*

R. Maarfi, E. L. Brown and S. Ramaswamy

Software Automation and Intelligence Laboratory, Department of Computer Science

Tennessee Technological University, Cookeville, TN 38505

Phone: (931)-372-3691. Email: srini@acm.org / srini@ieee.org

*ABSTRACT:* - This paper presents an Automated Highway Simulation using a hierarchy of communication to resolve advanced traffic situations. The simulation is being done to study communication paradigms, traffic concepts, and road design. The simulation is three dimensional; projected in a two-dimensional space, and implemented as a client/server and peer-to-peer system using Java. Vehicles communicate with each other using a linked graph called a 6 way DSM, or dynamic socket mesh. Cars also use a higher communication layer - called a segment controller, and segment controllers coordinate with each other using the simulation controller.

## I. INTRODUCTION

This paper presents the simulation of an automated highway system with intelligent vehicles using higher-level, off-road intelligence to aid in their guidance. This simulation is being designed to study communication paradigms, road design, and self-correcting systems. The concept of an automated highway system, or AHS, is nothing new. An AHS is a system that allows vehicles to drive themselves without human intervention. General Motors Corporation first introduced AHS in the 1939 World Fair. This system, and others that have followed, have made necessary the complete absence of human drivers. This simulation does not have that limitation. The main objectives of an automated highway system are not unlike any other system. The system should do its job as efficiently and as safely as possible while trying to take all errors that can occur into account. The system is designed to work around errors that may otherwise prevent it from functioning.

It is important to mention that a few key concepts have emanated from a four-layer protocol used in Berkley's PATH laboratories, funded by the Federal Highway Authority. The three-tier communication system discussed in this paper is designed to anticipate errors that may occur inside any part of the communication system itself, and will eventually be aware of mechanical failures that can occur and give the vehicle or the driver instructions. If any part of the communication system should fail, the remaining parts will change to a different operating mode to compensate for the failure. Each layer of the three-tier system is necessary for full functionality, but any layer can be removed during the simulation without catastrophic effects.

Platooning is the most obvious method available to increase road traffic density. A platoon is a group of vehicles with just enough space between them to react to each other, possibly even moving at predefined speeds. Previous systems, however, have relied on slightly slower speeds and closer vehicle proximity for a highway utilization increase of four times what human drivers use today. This increases the capacity of the highway by properly utilizing available space. Platoons are isolated from each other, and from human drivers, by large gaps. Earlier studies have shown that platoons can be created without having an adverse effect on safety. In this simulation, since we allow human-controlled cars, platoons are restricted. This control is a parameter of the simulation controlled by the second tier. Cars will then stay at a two-second following distance and switch lanes to maneuver around vehicles when necessary. With other humans around, the cars can be made to follow the normal rules of the road.

An AHS is one of the most appropriate applications to test a design for a self-correcting system because of the unpredictability factors, specifically humans and mechanical failure. Since human-controlled cars are allowed in the system, there is even greater unpredictability. Weather can seriously alter the performance of a vehicle also, but this is not currently added to the simulation. Communication layer failure is already implemented.

The simulation is implemented as a client/server networking application, written in Java. Vehicle to vehicle communication is allowed via peer-to-peer network connections, and communication between the segment controllers, the simulation controller and the vehicles are handled by client-server methods. This networking environment allows for a suitable way to test communication protocol and efficiency. The system was designed for efficiency in terms of processor utilization and network utilization. Each layer is handled at the lowest level possible, thus minimizing the need for passing information around the network. This is illustrated in Figure 1. The communication architecture is divided into three layers:

- i. Vehicle communication – Possibly representing sensors or radio communication, this is a way for cars to get information about the immediate surrounding cars such

as speed and position. In the simulation this is implemented by means of a linked list structure, the dynamic socket mesh or DSM.

- ii. Segment Controller – Segment controllers are local to specific sections of the highway. These sections are predefined internally, or are given to each segment controller by the simulation controller. Each vehicle only needs to communicate with the segment controller unless it is unavailable, then reaching the simulation controller is necessary. The segment controller informs a car of a malfunction in the six-way DSM.
- iii. Simulation Controller – Responsible for assigning sections of the highway to segment controllers. The simulation controller may also act as a segment controller if a segment controller is unavailable. Future plans involve routing of traffic during construction or accidents.

The paper is organized into four sections. In section II, the vehicle communication, segment controller and simulation controller are described along with their respective issues. In section III, the details of implementation are discussed. Section IV concludes the paper.

## II. DESIGN ISSUES

In our research, we have noticed that some limited intelligence need to be available to the vehicles themselves. Intelligent vehicles are a solution to the overhead normally associated with previous roadside-control design bottlenecks. In order for vehicles to behave intelligently, they must first be able to communicate with each other and share some required information. This section describes the design architecture for that communication and the application developed as a test-bed for evaluating performance. We assume three kinds of cars to be available in the simulation; viz. (i) Computer-Controlled Cars: These cars are the “smart” cars that can be controlled by the simulation and demonstrate the highest degree of intelligence. (ii) Human-Controlled Cars: These cars are controlled by human operators from a terminal and are completely unpredictable – similar to any highway scenario. (iii) Traffic Cars: These are “dumb” cars on the highway that just cruise along to create traffic situations – they are not controlled by any intelligent decision making process.

For our work the following three communication paradigms are readily applicable. The first, autonomous resolution; relies on receiving immediate data from the surrounding vehicles. This is a very limiting paradigm because in real-world situations vehicles may be unable to successfully request surrounding vehicles to even change lanes. The second approach, using a master/slave resolution, is not used unless a car is in a platoon. The

platoon leader is then assumed to be the master, and directs computer-controlled cars in its platoon similar to the way a traffic cop directs traffic.

The third paradigm, the mutual resolution paradigm, allows vehicles in close proximity to switch lanes without running into each other and to allow cars to switch into the desired lane in heavy traffic. This paradigm is used in our simulation for resolving the majority of highway maneuvers, however, application of a particular paradigm depends on the environment in the simulation. For example, if a human controlled car is in front of a computer-controlled car, the human controlled car cannot be asked to move out of the way or change speeds. A computer-controlled car is also unable to switch lanes, unless it is able to get clearance, which again may not be possible with a human-controlled car in the way. Mutual resolution, however, would allow a computer-controlled car to switch lanes first if, for example, it needed to reach the exit that would otherwise be blocked. A central controller is not necessary for this situation if each car is able to communicate its current goals to the other car, and each car is aware of what to do depending on the adjacent cars’ (shared information, if available) goals, its goals, and its surroundings. This is similar to a finite state machine. Each car in a normal situation makes its decisions based on a predetermined coordinating logic.

There are advanced traffic situations involving nonstandard vehicles, such as ambulances, roadblocks and closed lanes. Using sensor data (simulated data this implementation), a car may be able to see a blocked lane or a stopped car, but the delay in seeing the block and being able to switch lanes may cause significant traffic blockage.

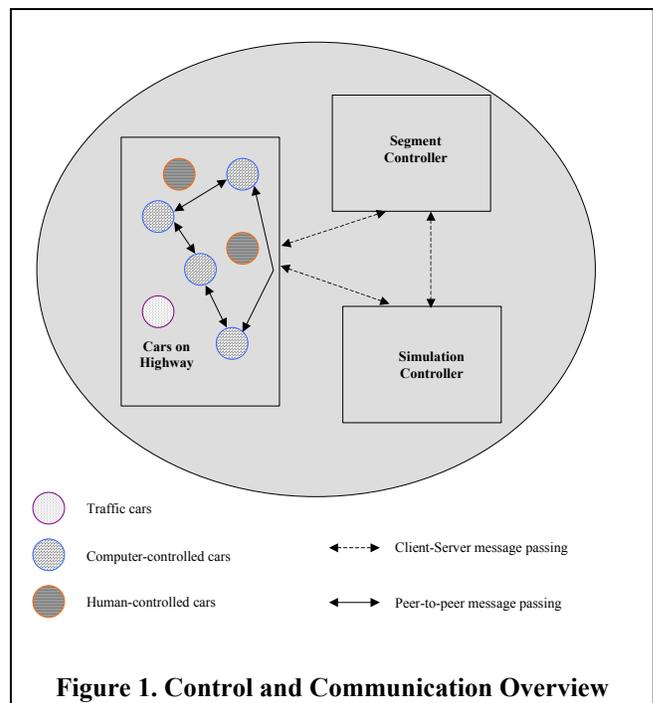


Figure 1. Control and Communication Overview

The same case applies with an ambulance, as shown in Figure 9 and Figure 10. Sensor data will allow a car to identify a vehicle and its speed, but it would be unable to switch lanes if another vehicle were to be in the way. While the implementation is currently not present, a car might also be advised to take a detour if an accident has occurred or road construction prevents passage. These and similar tasks are handled by the segment controller. Each segment controller is responsible for an arbitrarily large section of the highway. The segment controller also informs cars of the

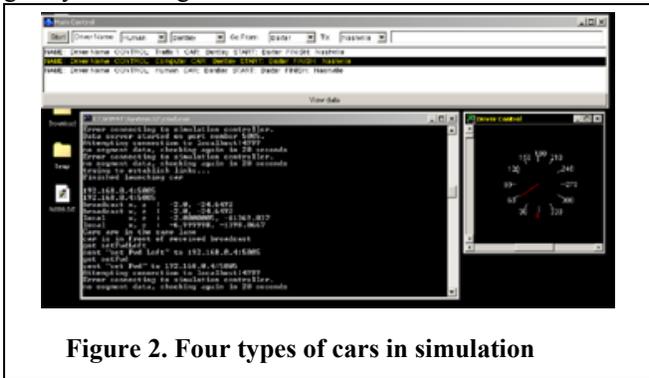


Figure 2. Four types of cars in simulation

existence of human drivers for resolving the decision to platoon and the speed at which to travel.

Since a segment controller only oversees a relatively small, portion of the AHS, a higher communication level is necessary for the controllers to be aware of unfavorable occurrences down the road, such as weather, traffic, or road construction, between segments and to assign segments of the highway to their respective controllers. This task is given to the simulation controller. Without the simulation controller, vehicles are unaware of intelligent decision making to decide upon the most efficient route to their destination. Besides, the simulation controller also is in-charge of initially setting up the necessary parameters of the simulation.

Each level of communication, however, is designed to function in the absence of the other. For example, if a car’s sensor should fail, the segment controller will be aware of the discrepancy between what one vehicle is reporting and what another vehicle is reporting. Once aware, the segment controller is capable of guiding a car to the nearest exit by relaying information that vehicles are reporting through the segment controller or, depending on the existence of humans in the segment, merely instruct the car to pull over or for the human driver to assume control. If a segment controller should fail, the simulation controller has the ability to function as a temporary segment controller. The vehicles attempt communication with the simulation controller automatically whenever failure with the segment controller occurs. This also allows for segment controllers to be changed without affecting the simulation. If both the simulation and segment controllers should fail, the vehicles will still be capable of navigating the highway, but they will

be unable to platoon since the existence of human drivers is unknown. They will be unaware of advanced traffic situations.

As shown above in Figure 2, three control types are available. Traffic 1 represents a car that is unaware of its environment and is only there to be an obstacle to other vehicles. Traffic 1 never makes a lane change, however, Traffic 2 randomly changes lanes. It will not change lanes into other cars, but provides an advanced challenge to the computer-controlled cars. The computer-controlled car is capable of joining or leading platoons, and it is capable of intelligently navigating around traffic and human-controlled cars. Human-controlled cars are completely driven by an operator at the terminal. The car controls are shown in the mid-right section and are capable of allowing lane changes or speed changes. The speedometer shows the current speed, and the bottom scrollbar shows the current lane.

III. IMPLEMENTATION

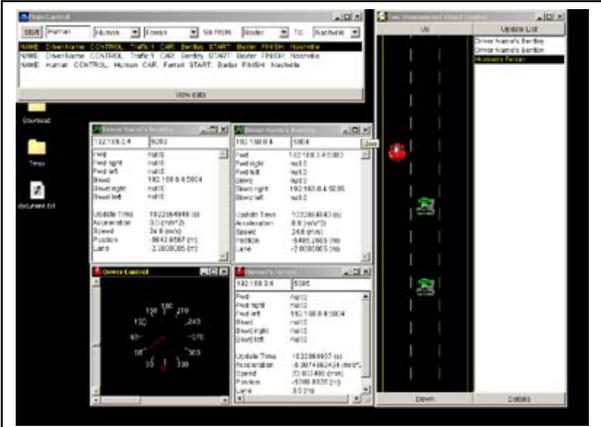


Figure 3. Links and Visualization

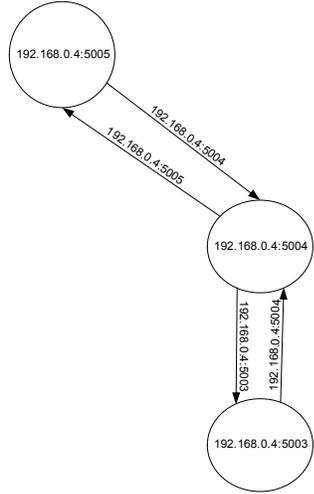


Figure 4. Internal representation of links

Effective networking, avoiding delays and redundancy, is critical to this project because of the representation of vehicle communication between each other, between segment controllers and between simulation controllers. To simulate real-world radio/wireless communications, the simulation uses TCP/IP sockets to communicate with each other. Necessary sensors are also represented as sockets in the implementation.

Inter-vehicle communication is represented as a series of six sockets, called a six-way dynamic socket mesh, or DSM. Each direction represents what the vehicle would see at a given location, specifically forward, forward-right, backward, backward-right, backward-left, and forward-left. This is a linked structure that forms a graph, and each car is a point on the graph. This represents sensory information gathering in a real-world implementation, so they are able to get basic operational data from these connections. However, using the mutual resolution paradigm allows for cars to communicate with each other to exchange awareness of their current goals – this will need more than just sensing capabilities; i.e. intelligent message passing, if implemented in a real-world scenario. However, to mimic expected reality, this is assumed to only happen between two computer-controlled cars.

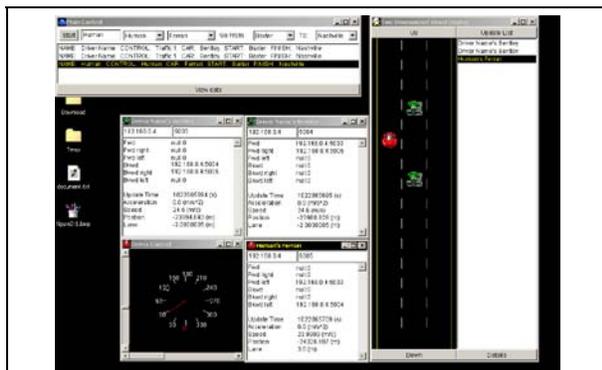


Figure 5. Passing the first green car

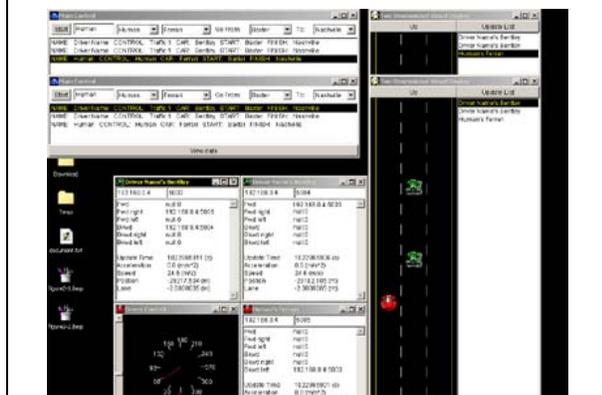


Figure 6. Passing the second green car

Automatic testing programs were written to ensure proper functioning of the simulation. In addition to testing programs, a rudimentary visualization was implemented to analyze the simulation. Java was the language of choice for the implementation, since it aids in portability and simplifies the distributed networking implementation. Figure 3 and Figure 4 demonstrate how the links are represented. The right hand side of Figure 3 shows the visualization application.

The dialog boxes shown in Figure 3 surrounding the speedometer represent the data stored by each car, which can be retrieved by double clicking on either the individual listing of the car in the main control application or by double clicking the list in the Visualization. This shows that each car identifies and locates others by storing IP addresses and sockets. Therefore, in the simulation, computer-controlled cars have the same link maintenance logic as human and traffic. The green cars in Figure 3, (and Figure 5, Figure 6) are traffic cars while the red car is a human-controlled car. It is to be noted that in the visualization, cars are traveling downward, so in this case the forward link is actually down, right is left, vice versa. In Figure 4, the circles represent cars and their unique nodes on the network, and the arrows pointing to other cars represent the connections on the graph. This concludes that a link is represented by two TCP/IP socket connections, which is similar to sensors mounted on the two cars.

In Figure 5 and Figure 6, the continuing validity of the links is illustrated. It shows that as the red car passes the first green car, it loses its forward right connection to it and gains a backward right. Conversely, the green car gains a forward left and loses a backward left. The front green car gains a backward left. Note that the backward left link was invalid to have by definition of the graph until the first green car had been passed.

As mentioned earlier, the segment controller is in charge of all vehicles within a small portion of the highway. When a segment controller is launched, it connects to the

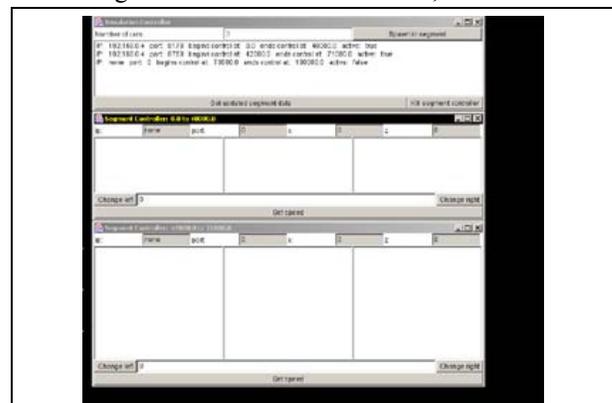
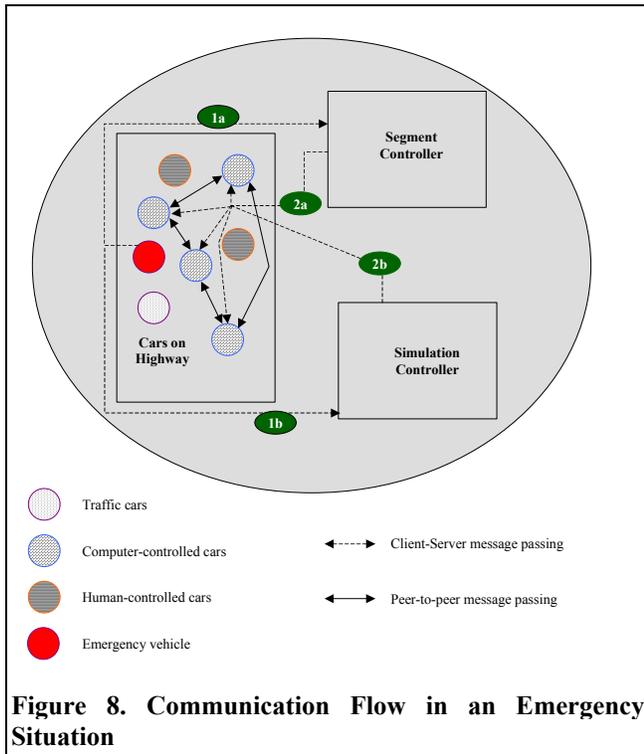


Figure 7. Simulation and segment controllers

simulation controller to determine what area of influence it has. If the simulation controller is unavailable, it reads from a file what section it is supposed to be controlling. This allows functionality without the simulation controller, but allows the simulation controller to assign new areas if change is necessary. Unlike the segment controller, each vehicle first attempts to connect its last known segment controller for the area it is currently located. If no segment is available, the car then attempts to connect to the simulation controller to locate another segment controller. If the simulation controller does not have a predefined segment for the vehicles location, the car is informed. If a segment is predefined but a controller does not exist, the simulation controller launches a segment controller for it to connect to. This means that the simulation controller actually runs segment controller logic if a segment is absent, and tells the car to connect to it.

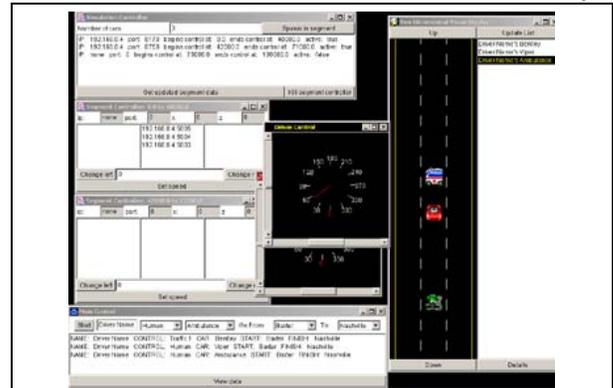
Figure 7 shows two segment controllers executing and the simulation controller at the top of the screen. The three lines (of text) at the top of the simulation controller represent a predefined segment. The first two are active, and the last is inactive. The three white areas in the middle of the segment controllers (no cars are located in either segment) represent each lane of the highway. The change right and change left buttons allow for the movement of cars



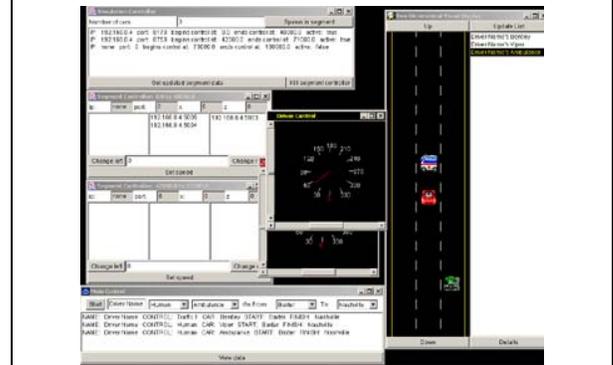
by the segment controller for testing purposes, as does the set speed.

A useful application of the simulation controller can be seen in Figure 8. Figure 9 and Figure 10 show the

corresponding implementation for the scenario depicted in Figure 8. Here, a computer-controlled vehicle – an ambulance, gets priority in using a lane and all computer-controlled cars are instructed to move to a different lane. The ambulance will use either the segment controller or the simulation controller to clear its lane. In the case illustrated in Figure 9, an ambulance vehicle has just been informed of an emergency, but two cars (a red human controlled car – immediately upfront and a green computer-controlled car before the human controlled car) are in the way. The human-controlled car (line of sight communication) may not immediately move over while the other car is a computer-controlled car. Since the ambulance is unable to directly see



**Figure 9. Ambulance wishes to pass**



**Figure 10. Ambulance can only move computer cars out of its way**

the computer-controlled car, it would be unable to inform the car to move out of the way. However, with the segment controller, the ambulance sends a message to the segment controller, which, in turn, sends a message to every computer or traffic-controlled car in the lane, within a predefined range, commanding it to move out of the way. Figure 10 shows the segment controller’s ability to request the green car (computer-controlled) out of the ambulance’s way, even though the human car (one right before the ambulance in the middle lane) is blocking its view.

The segment controller increases network efficiency in cases with more cars because each node on the graph does not have to be traversed to relay messages. However, a computer-controlled car would only be able to dodge the ambulance if it was directly in front of it.

More importantly, a lane lock has occurred in the middle lane. Set by the segment controller, computer controlled cars are unable to switch back into the middle lane until the ambulance is out of the way. To increase network efficiency, vehicles are unaware of the lane lock until attempting to switch into the lane. This allows cars to switch lanes without informing the segment controller. Without the segment controller, cars would have to check their diagonal links for situations like an ambulance, but due to the functionality of the links demonstrated in Figure 3, Figure 5, and Figure 6, they might still not have seen the ambulance coming until it began to switch lanes.

#### **IV. CONCLUSION**

This paper presented the implementation of a flexible and robust three-tier communication and control structure applied to a distributed simulation of an Automated Highway Simulation. Vehicles can be controlled by humans or be logic driven. This paper captures the most likely scenario of any future AHS as a result. The Java platform provides adaptability and object-oriented structure that allows for modularization for easier future modification and maintenance, and allows portability. The paper has also shown that while external communication is not necessary for functionality, efficiency is greatly increased and safety in the face of failure is ensured.

#### **V. REFERENCES**

- [1] R. E. Fenton and R. J. Mayhan, "Automated Highway Studies at Ohio State University- An Overview," *IEEE Trans. on Vehicular Technology*, vol. 40, no. 1, pp. 100-113, Feb. 1991.
- [2] C. Unsal, P. Kachroo and J. S. Bay, "Multiple Stochastic Learning Automata for Vehicle Path Control in an Automated Highway System," *IEEE Trans on Systems, Man and Cybernetics*, vol. 29, no.1, pp. 120-127, Jan. 1999.
- [3] D.N. Godbole, J. Lygeros, E. Singh, A. Deshpande, A.E. Lindsey. "Communication Protocols for a Fault Tolerant Automated Highway System". Dept of Elec. Engg. and Comp. Sciences. Univ. of CA.
- [4] R. Donath, April Dean, D. Girardi and S. Ramaswamy, " Distributed Simulation of an Automated Highway System with Intelligent Vehicles ", *2001 Summer Computer Simulation Conf*, July 2001, Orlando, Florida.
- [5] Luke D. Postema, "Automated Highway Systems: Incremental Deployment as a Solution for The Future of Transportation?". Washington internships for Students of Engineering. August 1998