A Simulation and Design System for Tactical Driving Algorithms

Rahul Sukthankar, John Hancock, Dean Pomerleau, Chuck Thorpe
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3891
Internet: {rahuls—jhancock—pomerlea—cet}@ri.cmu.edu

Abstract

Intelligent vehicles must make real-time tactical level decisions to drive in mixed traffic environments. Since repeatable testing of different algorithms in rare and potentially dangerous situations is necessary, we have developed a custom simulator for this task. SHIVA (Simulated Highways for Intelligent Vehicle Algorithms) mirrors many aspects of the Carnegie Mellon Navlab [26, 13] system, enabling algorithms developed in simulation to be implemented on the robot with minimal modification. Realistic sensor models encourage developers to create algorithms that will work on real robots. Incremental development is facilitated through hierarchies for vehicles, sensors and reasoning objects. An integrated simulation and animation environment provides interactive graphical debugging capabilities.

Introduction

Simulators are often used to aid in the design of complex systems that are too difficult or expensive to prototype or too dangerous to test in reality. While these simulators may be helpful in validating a design, they are generally not useful as design tools in themselves. We have developed a system, SHIVA (Simulated Highways for Intelligent Vehicle Algorithms), for simulation and design of tactical driving algorithms.

In this paper we focus upon SHIVA's role as a design tool in the development of intelligent vehicle algorithms. The internal structure of the simulator is more comprehensively covered elsewhere [25].

1 Tactical Level Driving

The driving task can be characterized as consisting of three levels: strategic, tactical and operational [18]. At the highest (strategic) level, a route is planned and goals are determined; at the intermediate (tactical) level, maneuvers are selected to achieve short-term objectives such as deciding whether to pass a blocking vehicle; and at the lowest level, these maneuvers are translated into control operations.

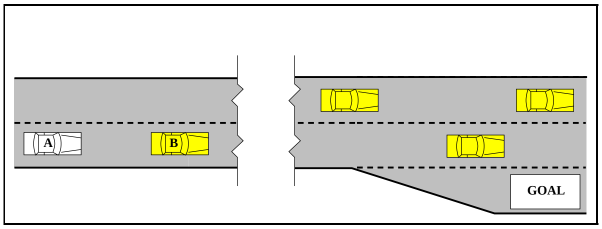


Figure 1: Example of a tactical scenario.

Mobile robot research has successfully addressed the three levels to different degrees. Strategic level planners [2, 7, 14, 22, 28] have advanced from research projects to commercial products. The operational level has been investigated for many decades, resulting in systems that range from semi-autonomous vehicle control [4, 9, 10, 16, 17, 19] to autonomous driving in a variety of situations [6, 15, 27, 23, 20]. However, the decisions required at the tactical level are difficult and a general solution remains elusive.

Consider the typical scenario depicted in Figure 1: Our vehicle (A) is in the right lane of a divided highway, approaching the chosen exit. Unfortunately, a slow car (B) blocks our lane, preventing us from moving at our preferred velocity. Our desire to pass the slow car conflicts with our reluctance to miss the exit. The correct decision in this case depends not only on the distance to the exit, but also on the traffic configuration in the area. Even if the distance to the exit is sufficient for a pass, there may be no suitable gaps in the right lane ahead before the exit. Thus tactical level reasoning combines high-level goals with real-time sensor constraints in an uncertain environment. The SAPIENT[24] project addresses these problems and aims to drive the Carnegie Mellon Navlab [26, 13] vehicles in real traffic.

Simulation is essential in developing such systems because testing new algorithms in human traffic is risky and potentially disastrous. SHIVA not only models the elements of the driving domain most useful to designers but also provides tools to rapidly prototype and test algorithms in challenging traffic situations.

2 Overview

SHIVA is a kinematic simulation of vehicles moving and interacting on a user-defined stretch of roadway. The vehicles can be equipped with simulated human drivers as well as sensors and algorithms for automated control. These forms of vehicle intelligence influence the vehicles' motion through simulated commands to the accelerator, brake and steering wheel. SHIVA's user interface provides facilities for visualizing and influencing the interactions between vehicles.

The tactical-level focus in SHIVA directs a number of choices. We do not model vehicle dynamics since their effects are primarily at the operational level. Additionally, since the number of vehicles in a given scenario is rather small, we can simulate detailed sensor models for the vehicles in close to real-time. Providing simultaneous simulation and visualization lets researchers interact with the vehicles and encourages incremental development of reasoning systems.

3 Incremental Development

Most existing simulators are designed to model only a few vehicle configurations and reasoning agents. By contrast, SHIVA's architecture is open-ended enabling researchers to integrate new sensors, controllers and intelligences into existing vehicle specifications. Researchers can study interactions between different intelligent algorithms by creating scenarios with a variety of vehicle configurations. An example of this is mixed-mode traffic where human operated and autonomous vehicles use very different sensors and driving strategies. However all vehicles can be functionally represented as consisting of three subsystems: perception, cognition and control. A view of this architecture is shown in Figure 2.

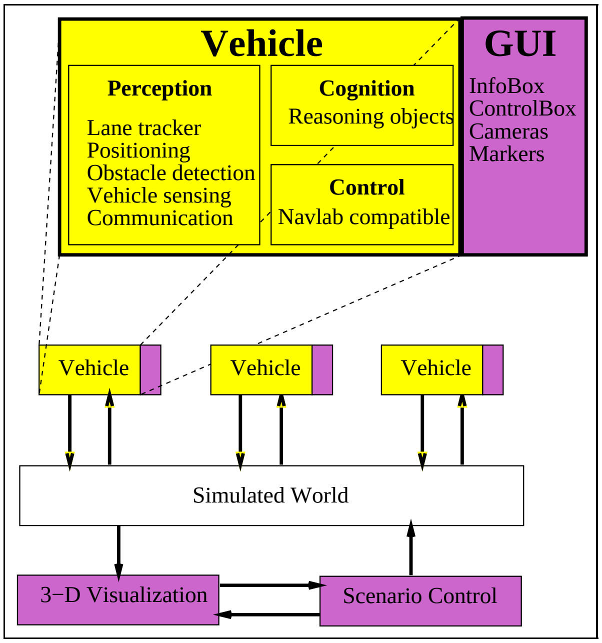


Figure 2: SHIVA architecture

3.1 Perception

Largely ignored in most simulators, perception remains one of the most difficult problems in mobile robotics. Control algorithms which make unrealistic perceptual assumptions are destined to remain unimplementable on real systems. On the other hand, perfectly modeling realistic sensors is infeasible since the simulated world cannot match the complexity of real life. Simulator designers must therefore select an appropriate level of detail for their task. In the tactical driving domain, issues such as occlusion, ambiguity and obstacle-to-lane mapping are important; SHIVA supports a variety of sensors and perception routines that enable cognition algorithms to gather information about their surroundings in a realistic manner. Perception objects are grouped into an open-ended sensor hierarchy (See Figure 3) which allows designers to create appropriate models and swap them into existing vehicle configurations with minimal modification.

A typical vehicle configuration contains sensors from the following list:

- Positioning: (dead-reckoning and/or GPS)
- Lane tracking: (possibly with exit ramp counting features)
- Car tracking modules or rangefinders.

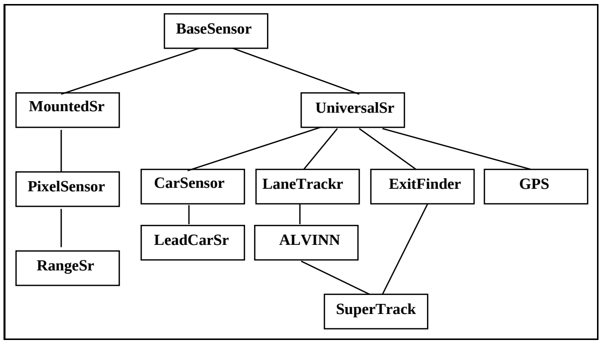


Figure 3: Subset of SHIVA's sensor hierarchy

- Self-state: odometer, speedometer
- Optional: communication (for communication with other equipped vehicles)

Most of the sensor modules are functionally equivalent to their real counterparts—that is, they provide a similar interface without simulating failure modes or peculiarities which depend on unmodeled phenomena (for example, GPS satellite tracking problems or dead-reckoning wheel slip). The outputs of the sensors can be corrupted by noise if desired.

Some sensors are explored in greater depth—vehicle detection in particular, since it is of critical importance at the tactical level. Designers may select from two main types of vehicle sensors. The first is a sweeping rangesensor that returns an array of pixels (corrupted by noise in angle and range). The sensor is usually scanned on demand (enabling selective perception) and certain rangesensors may be actively panned to examine particular regions of interest. The challenging tasks of detection, segmentation, classification, tracking, velocity estimation and object-to-lane mapping are all left to the cognition module. These tasks are often so difficult (and computationally expensive) that designers also need a more abstract vehicle sensor that can be used to prototype reasoning schemes. Thus, the second sensor model assumes that all of these tasks can be reliably performed and directly outputs the position and velocity of the relevant vehicles. Higher order information such as acceleration of the detected vehicle is not available since real sensors do not directly return that data.

Robust sensing modules coupled with well-tuned controllers are generally sufficient for operational level tasks such as road following or emergency braking. However such systems are inadequate for driving in traffic since higher level reasoning is required.

3.2 Cognition

Situational awareness is the key to effective navigation in traffic. Intelligent vehicles must assess the outcomes of various actions and balance the desires of higher-level goals (such as taking a specified exit) with sensor-driven constraints (e.g. observed vehicles). Finally, these algorithms must be tested under a variety of dangerous situations before they are implemented on a real robot. SHIVA provides a flexible architecture for vehicle intelligence enabling designers to simulate different configurations of sensors and reasoning systems simultaneously. Since perception and cognition are interconnected (especially with active sensing), SHIVA encourages designers to group compatible cognition and perception modules into a vehicle configuration.

We implemented two radically different reasoning schemes to evaluate SHIVA as a design tool. The first vehicle configuration was a classical (hand-coded) rule-based expert with selective perception. The second approach employed a distributed reasoning object [1] [24] architecture with a voting arbiter. Inspection and control tools for the two types of vehicle were specialized from the defaults provided.

The test track for these experiments is shown in Figure 4. Vehicles were instructed to go around the course for a specified number of laps before taking the exit. The design of the track ensured that the ramps are always congested and that vehicles had to maneuver through considerable traffic to complete the task successfully. Note that this track is functionally similar to a straight highway with a series of alternating on- and off-ramps; however the circular geometry recycles vehicles and creates similar traffic densities with fewer total cars (allowing us to simulate the scenario in real time). An additional benefit of the cyclotron track is that we may view the entire scene at once.

Each reasoning scheme posed different challenges for SHIVA. The expert system rules had to be debugged in specially crafted scenarios; this was accomplished using the ControlBox tools (See Section 4.3). The reasoning object system required fine-tuning of many parameters (each associated with a particular reasoning object); this was enabled through a set of slider bars.

After experiencing the difficulty of hand-tuning the reasoning object parameters we are currently investi-

---

[1] Each reasoning object makes maneuver recommendations based upon a single element in the world (e.g. upcoming exit or blocking vehicle) and returns votes over the common action space.

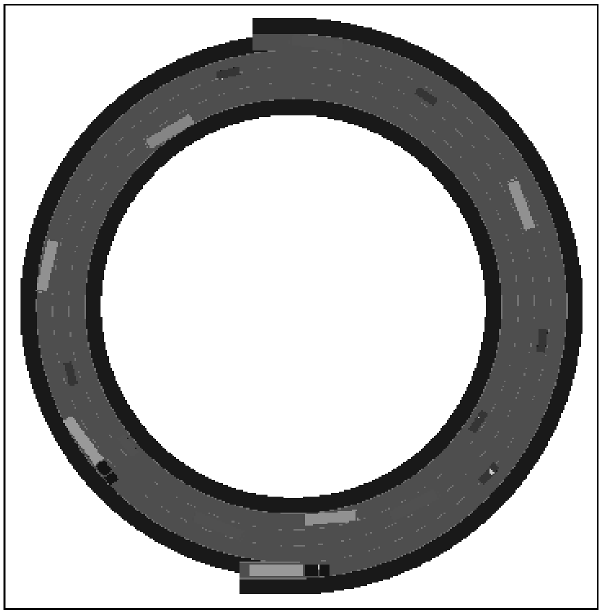


Figure 4: Cyclotron test track

gating learning techniques [1] to automate this task. Potential solutions are tested on a series of challenging scenarios which SHIVA auto-loads for every candidate vehicle.

3.3 Control

This module provides an interface to the simulated actuators that drive SHIVA's robot vehicles. The commands accepted by the controller are identical to those available on the Carnegie Navlab robots (i.e. desired velocity and steering curvature). This compatibility ensures that algorithms developed in SHIVA can be ported to the robot vehicle with few changes.

4 Simulation Environment

Simulations are primarily used to validate a design. Since reasoning in tactical situations is complex, algorithms are best developed in an iterative manner. This introduces a second need: the ability to rapidly refine the reasoning systems. Tools necessary for a useful simulation and design system fall into three main categories.

4.1 Visualization/Validation Features

The first step to validating algorithms is observing them in action. Visualization tools allow the designer to qualitatively evaluate his/her algorithms from different points of view. To verify whether vehicles are behaving reasonably with a given design, the researcher needs to be able to see how the vehicles behave both as individuals and as aggregates.

SHIVA provides a flexible suite of visualization tools (see Figure 5) using Open Inventor (a 3-D graphics library developed by Silicon Graphics). SHIVA's primary view is an interactive camera which shows the simulation from overhead at an aggregate level. Multiple views that track vehicles may be created by selecting the desired vehicle(s) in the primary view. These secondary views can display driver's eye (or other arbitrary) perspectives. Since humans make tactical decisions from behind the wheel, these views are helpful in judging the quality of decisions made by the AI algorithms. A vehicle selected for monitoring can be ordered to change color or begin dropping virtual bread-crumbs allowing us to observe not only the vehicle's current position, but also its previous trajectory.

SHIVA's visualization tools also support a host of features such as dumping views to PostScript (for printing), scenario control and creating VRML (Virtual Reality Modeling Language) files for Webspace browsers.

4.2 Analysis Features

To improve on current algorithms, we need the ability to analyze what the vehicles are doing correctly, and to identify what they are doing incorrectly. Although visualization tools allow the designer to see if something is wrong with the algorithms, this information is generally insufficient to diagnose the problem. To perform this quantitative analysis on-the-fly, researchers require access to reasoning object internals during the simulation. Current debugging tools are inadequate for this task since they only display a few variables at a time.

SHIVA displays qualitative and quantitative information through InfoBoxes which are continually updated during the simulation. Researchers may request this information on a per-agent basis and focus only on the relevant details. Since simulation and animation is fully integrated, problems spotted using the visualization tools may be immediately investigated. Most importantly, these InfoBoxes can be easily customized for each vehicle configuration, so that all of the relevant information is displayed. In cases where no customization is done, InfoBoxes display whatever information they can (generally information provided by super-classes of customized objects).

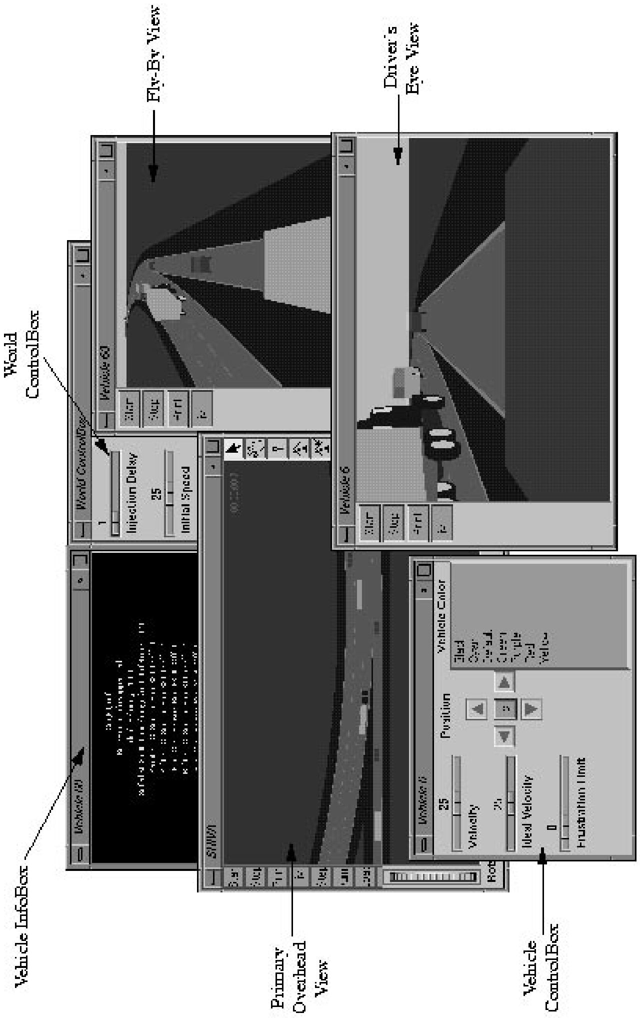
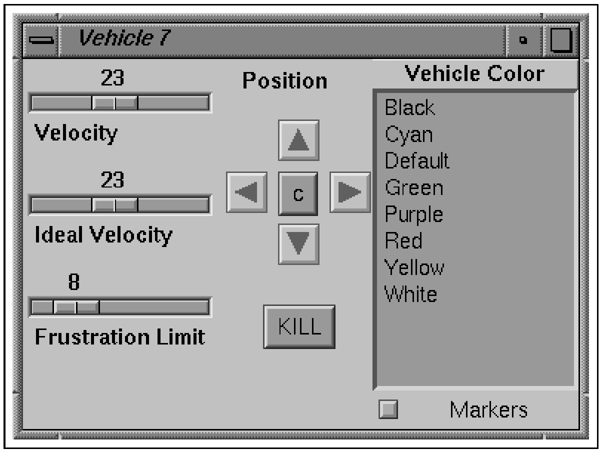


Figure 5: SHIVA desktop



23
Velocity

23
Ideal Velocity

8
Frustration Limit

Figure 6: Example of a ControlBox

Global information is summarized in a WorldInfoBox that enables researchers to monitor statistics such as average speed and vehicle exit success rates. The analysis tools are open-ended, encouraging customization for specific scenario needs. Furthermore since vehicle classes are organized in a hierarchy, new designs can inherit most of their essential debugging elements from pre-existing tools.

4.3 Interaction/Modification Features

The final important requirement for a simulation and design system is the ability to interact with and modify objects on-the-fly. This provides several benefits:

1. Supports rapid iterative development.
2. Enables better exploration of parameter space.
3. Allows fine-tuned scenario control.

Iterative development without recompiling is vital for incremental algorithm design. Interactive parameter modification can be used to expose algorithms to sudden changes in environment, encouraging robustness. This is done primarily through SHIVA's ControlBoxes hierarchical interface tools which can be customized for particular vehicle configurations (See Figure 6). By enabling researchers to interactively create traffic situations on the fly, ControlBoxes also provide a means for scenario control.

5 Scenario Control

Many interesting traffic situations occur very rarely in both real life and in simulation. However these are also the scenarios where tactical level reasoning systems are most challenged. Rather than forcing researchers to wait patiently for these events to arise, SHIVA provides users with several interactive scenario control features.

By default, ControlBoxes allow users to change basic parameters of the selected vehicle such as current velocity and position. The latter is adjusted through a tactical driving interface where controls are mapped onto the instantaneously local road coordinate frame. This automatic re-orientation of the vehicle makes scenario control straightforward even on complicated road structures. Researchers may add other features to the existing class to facilitate changing of parameters. For example, the ControlBox in Figure 6 provides slider bars for some of the key parameters in that vehicle configuration. Since SHIVA offers both Pause and Step options in addition to Play, researchers can easily use ControlBoxes to affect the current simulation state to create the appropriate scenario. Interesting scenarios or simulation states may then be saved to disk for future use.

Additionally SHIVA provides ways to influence the behavior of vehicles in simulation. A breakdown function allows users to selectively stop vehicles (the target vehicle is forced to brake as hard as possible until it comes to rest).

6 Saving/Restoring Simulation State

Some simulations [12] are built on top of a persistent database. This enables them to save and restore the simulation state as needed. Although SHIVA is not implemented on such a database, subsets of this feature have been made available. Users may interactively save the current state of the simulation to file for later examination. Since the state is stored as a commented ASCII file, users are able to see (and edit) any of the elements in the simulation. The primary purpose of this feature is to enable researchers to interactively create traffic scenarios and then efficiently simulate them off-line without graphics. SHIVA can also dump state at regular intervals for later visualization or analysis. This feature also allows researchers to observe the evolution of the system from the same initial conditions given different reasoning object parameters.

7 Comparisons with existing tools

While microscopic ground vehicle simulators have been in use for over thirty years [3, 5, 11, 29], no existing tool has all the capabilities desirable for designing intelligent vehicle algorithms. Simulators are generally created to study traffic patterns and consequently fail to provide adequate facilities for adding or debugging new vehicle control algorithms.

Three simulators address tactical-level modeling of intelligent vehicles: Pharos [21], SmartPath [8] and SmartAHS [12]. Pharos focused on important perceptual issues and directed SHIVA's early development. SmartPath is well suited to modeling the PATH AHS concept (even with large numbers of vehicles) and its SGI animation package influenced some of our visualization tools. SmartAHS is an object-oriented simulator which stores its evolution state in a persistent DBMS and motivated SHIVA's state saving/loading feature.

While all three simulators present valuable insights, none is completely suitable for developing tactical algorithms. Pharos has unrealistic sensor expectations and offers no scenario control features; SmartPath does not support reasoning systems which violate the layered state-machine architecture; SmartAHS (still under development) does not currently offer suitable sensor or reasoning object hierarchies, nor does it provide an integrated simulation/visualization environment encouraging iterative algorithm development. In fairness, it must be noted that tactical algorithm generation is not the primary aim of any of these simulators.

For tactical level interactions, fairly simple vehicle models are sufficient. Although vehicles are kinematically accurate with continuous lane changing and realistic steering characteristics, SHIVA does not model vehicle dynamics, road-surface interactions or collisions realistically. Therefore we feel that the current implementation of SHIVA is unsuitable for applications where these factors are important (e.g. closely-spaced platoons).

8 Future Work

To supplement the scenario control tools already available, we plan to build tools to simplify save-file editing. Since SHIVA has been designed to expand to fill the future needs of intelligent vehicle designers, we expect to develop additional display and control features as they become necessary. Projects in progress include: learning optimal reasoning object parameters using evolutionary algorithms [1]; investigating loosely-coupled cooperation between intelligent vehicles; creating human driver models; and developing a human driver interface for teaching supervised learning algorithms [20].

9 Conclusions

SHIVA is best suited for simulating small numbers of vehicles in detail rather than a large highway network. Our primary focus is to develop more intelligent reasoning systems within SHIVA to address the problem of driving in traffic. The cognition system SAPIENT [24] will be ported to the Navlab vehicles and driven in real traffic.

An interactive Webspace demo of SHIVA can be found on the World Wide Web at:
¡http://www.cs.cmu.edu/~rahuls/shiva.html¿

10 Acknowledgements

This research was partially sponsored by the Department of Transportation, under contract Automated Highway System (contract number DTFH61-94-X-00001).

The authors would like to thank Shumeet Baluja for his suggestions on evolutionary algorithms and for reviewing drafts of this paper.

References

[1] Shumeet Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report CMU-CS-94-163, Carnegie Mellon University, 1994.

[2] P. L. Belcher and I. Catling. Autoguide electronic route guidance in London and the U.K. Technical Report 89102, ISATA, June 1989.

[3] M. Booth, J. Cremer, and J. Kearney. Scenario control for real-time driving simulation. In Proceedings of 4th Eurographics Animation and Simulation Workshop, September 1993.

[4] K. H. F. Cardew. The automatic steering of vehicles an experimental system fitted to a Citroen car. Technical Report RL340, Road Research Laboratory, February 1970.

[5] H. C. Chin. SIMRO: A model to simulate traffic at roundabouts. Traffic Engineering and Control, 26(3):109–113, March 1985.

[6] E. Dickmanns and A. Zapp. A curvature-based scheme for improving road vehicle guidance by computer vision. In Proceedings of the SPIE Conference on Mobile Robots, 1986.

[7] R. J. Elliot and M. E. Leak. Route finding in street maps by computers and people. In Proceedings of AAAI, 1982.

[8] F. Eska, D. Khorramabadi, and P. Varaiya. SmartPath: An automated highway system simulator. Technical Report UCB-ITS-94-3, University of California, Berkeley, 1994.

[9] R. Fenton and R. Mayhan. Automated highway studies at the Ohio State University — an overview. IEEE Transactions on Vehicular Technology, 40(1):100–113, February 1991.

[10] K. Gardels. Automatic car controls for electronic highways. Technical Report GMR-276, General Motors Research Labs, June 1960.

[11] D. R. P. Gibson. The Application of Traffic Simulation Models, chapter Available Computer Models for Traffic Operations Analysis, pages 12–22. National Academy of Sciences, 1981. TRB Special Report 194.

[12] Alex Gollu. Object Management Systems. PhD thesis, University of California–Berkeley, May 1995.

[13] T. Jochem, D. Pomerleau, B. Kumar, and J. Armstrong. PANS: A portable navigation platform. In Proceedings of IEEE Intelligent Vehicles, 1995.

[14] H. Kawashima. Two major program demonstrations in Japan. IEEE Transactions on Vehicular Technology, 40(1):141–146, February 1991.

[15] Karl Kluge and Charles E. Thorpe. Explicit models for road following. In Proceedings of the IEEE Conference on Robotics and Automation, 1989.

[16] R. P. Lang and D. E. Focitag. Programmable digital vehicle control system. IEEE Transactions on Vehicular Technology, VT(28):80–87, February 1979.

[17] Ichiro Masaki, editor. Vision-Based Vehicle Guidance. Springer-Verlag, 1992.

[18] J. A. Michon. A critical view of driver behavior models: What do we know, what should we do? In L. Evans and R. Schwing, editors, Human Behavior and Traffic Safety. Plenum, 1985.

[19] R. Oshima et al. Control system for automobile driving. In Proceedings of the Tokyo IFAC Symposium, 1965.

[20] Dean A. Pomerleau. Neural Network Perception for Mobile Robot Guidance. PhD thesis, Carnegie Mellon University, February 1992.

[21] Douglas A. Reece and Steven A. Shafer. An overview of the Pharos traffic simulator. In J. A. Rothengatter and de Bruin R. A., editors, Road User Behavior: Theory and Practice. Van Gorcum, Assen, 1988.

[22] J. H. Rillings and R. J. Betsold. Advanced driver information systems. IEEE Transactions on Vehicular Technology, 40(1), February 1991.

[23] Rahul Sukthankar. RACCOON: A Real-time Autonomous Car Chaser Operating Optimally at Night. In Proceedings of IEEE Intelligent Vehicles, 1993.

[24] Rahul Sukthankar. Situational awareness for driving in traffic. Thesis Proposal, October 1994.

[25] Rahul Sukthankar, Dean Pomerleau, and Charles Thorpe. SHIVA: Simulated highways for intelligent vehicle algorithms. In Proceedings of IEEE Intelligent Vehicles, 1995.

[26] Charles E. Thorpe, Martial Hebert, Takeo Kanade, and Steven A. Shafer. Vision and navigation for the Carnegie Mellon NAVLAB. IEEE Transactions on PAMI, 10(3), 1988.

[27] M. Turk, D. Morgenthaler, K. Gremban, and M. Marra. Video road following for the Autonomous Land Vehicle. In Proceedings of the International Conference on Robotics and Automation, 1987.

[28] R. von Tomkewitsch. Dynamic route guidance and interactive transport management with ALI-Scout. IEEE Transactions on Vehicular Technology, 40(1):45–50, February 1991.

[29] Shui-Ying Wong. TRAF-NETSIM: How it works, what it does. ITE Journal, 60(4):22–27, April 1990.