Situational Awareness for Driving in Traffic

A Thesis Proposal

Rahul Sukthankar

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

e-mail: `rahuls@ri.cmu.edu`

October 31, 1994

Abstract

Situational awareness is the key to effective navigation in traffic. Although existing robot systems are capable of robustly performing many elements of the navigation task (e.g. road following and car tracking), integrating these aspects into a competent driving system requires a higher-level understanding of the driving domain.

I propose to build a reasoning system, SAPIENT (Situational Awareness Planner Implementing Effective Navigation in Traffic) that understands the tactical driving task. Defensive driving heuristics, occlusion reasoning, map-based reasoning and experience-based prediction of traffic flows will be captured in *reasoning objects*. Explicit representations of intentions (higher-level goals) will be used to create and destroy objects in response to changing situations. In a fully autonomous implementation, these intentions will be generated by higher level reasoning modules, while in the driver's associate application, they will be inferred from the driver's behavior. A loosely coupled architecture will enable SAPIENT to address higher level goals while reacting intelligently to sensor information.

SAPIENT will interact with existing modules such as road followers, car trackers and low level perception systems. Separate client modules will interface SAPIENT to higher level planners (both human and robot). I will build two representative clients: an autonomous driving module and a simple driver's associate.

A custom simulator with realistic sensor modeling will be used for development and SAPIENT will be tested on a set of driving scenarios extracted from real-life examples. Controlled experiments with the NAVLAB II will also demonstrate SAPIENT's competence in limited scenarios on real roads.

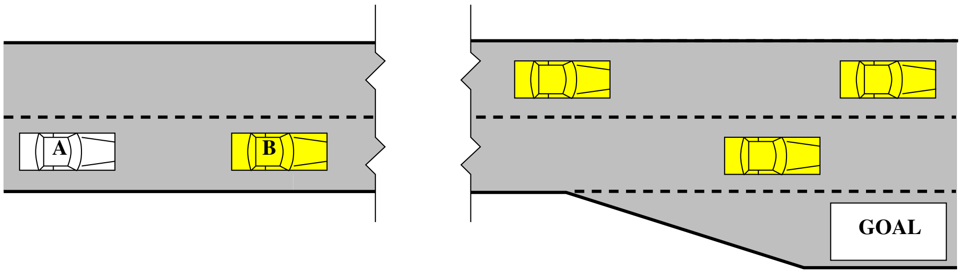

Figure 1: Example of a tactical scenario.

1. Introduction

The driving task can be characterized as consisting of three levels: strategic, tactical and operational [42]. At the highest level, a route is planned and goals are determined; at the intermediate *tactical* level, maneuvers are selected to achieve short-term objectives; and at the lowest level, these maneuvers are translated into control operations.

Mobile robot research has successfully addressed the three levels to different degrees. Strategic level planners [5, 16, 24, 55, 62, 75] 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 [9, 17, 20, 34, 38, 47] to autonomous driving in a variety of situations [15, 18, 28, 33, 46, 49, 60, 71, 72, 73, 76]. Some operational level behaviors have tactical level aspects — for example, a convoying module must be aware of road geometry since direct servoing will lead to corner cutting. Modules that address such tasks have also been successfully developed [6, 14, 19, 26, 31, 63, 64, 60]. However, the decisions required at the tactical level are difficult and a general solution remains elusive. Tactical plans are affected by a variety of factors including road geometry, traffic control devices, occlusion constraints and weather (e.g. reduced visibility from fog and icy roads in winter).

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. If we are familiar with the traffic along this route, or we are able to obtain information about traffic at the exit, an action may be chosen with greater confidence. Let us assume that our vehicle is equipped with two lower level modules: a lane-changing module, and a car-following module. Because the actions associated with *both* modules are locally feasible, an arbiter that selects an action based solely on the confidence of each module will be unable to choose between them. Since none of the low-level modules understand the scenario as a whole, a higher level reasoning is required. We call this *situational awareness*.

I propose to develop a system that implements situational awareness for tactical driving. SAPIENT (Situational Awareness Planner Implementing Effective Navigation in Traffic) will interact with existing lower-level modules including: road followers [49], obstacle detection modules [35], car followers [64] and strategic level planners [22]. SAPIENT will also process input from road-based sensors including measurements of traffic density, average velocity and map data. The qualitatively different situational awareness tasks will be implemented as *reasoning objects* which will execute asynchronously in real-time.

SAPIENT will be developed on a custom simulator with realistic sensor modeling and an implementation on the Carnegie Mellon NAVLAB II will demonstrate real-time performance in controlled environments.

SAPIENT's competence will be tested on a number of scenarios extracted from actual driving situations. Possible applications for SAPIENT include drivers' assistants, emergency countermeasure systems and tactical reasoning modules for robots.

2. Situational Awareness

Effective driving in traffic consists of:

1. Recognizing and reacting to potentially dangerous configurations.
2. Efficiently maneuvering through traffic.
3. Achieving higher level objectives while addressing lower level concerns.

The first objective involves detecting dangerous situations and generating good (though not necessarily optimal) actions. The second entails selecting lanes and speeds to optimize travel time while ensuring safety. The third requires modifying plans based on sensor information.

Situational awareness is the domain-specific understanding that enables drivers to achieve these goals. Knowledge about the environment (road geometry, weather effects on visibility), traffic (vehicle configurations, rate of flow), and driver behavior (our intentions, intentions of other drivers) are some important aspects of situational awareness. While the simpler safety concerns can be addressed by constant monitoring of nearby traffic, more complicated scenarios appear to require sophisticated reasoning and planning. In human drivers, experience is correlated with the development of situational awareness [82]. However the fact that inexperienced drivers are still able to drive suggests that situational awareness can be implemented incrementally. In SAPIENT, this is analogous to adding or modifying *reasoning objects* (See Section 4.2).

Driving in traffic is sometimes seen as the problem of dealing with obstacles in a dynamic environment. Difficulties of the domain include: real-time constraints on reasoning; incomplete sensor information (due to noise and occlusion); and unknown intentions of other drivers. Humans are able to drive effectively despite these challenges because they make good use of the available domain-specific knowledge — that the environment is highly structured (e.g. cars drive in lanes, roads curve gradually and meet at intersections etc) and that other vehicles usually behave in a reasonable manner. Furthermore since signs and lane markings are designed to be highly visible to humans rather than robots, the perception issues must also be addressed: car tracking and obstacle-to-lane mapping are examples of tasks which are easy for humans, but non-trivial for robots.

3. Related Work

Related research in tactical driving comes from a variety of fields: robotics, computer science, traffic engineering, automotive engineering and psychology. This section examines some work that has influenced SAPIENT's development.

3.1. Driver Models

The tasks and subtasks involved in human driving are comprehensively listed by Mc Knight and Adams [40]. Unfortunately, since their work is not directed towards computer implementations, many tasks are difficult to express computationally. Some recommended actions such as "[Driver] selects lane relative to car's speed, maneuvers and traffic flow" are too vague, while others are contradictory. Although human drivers may be able to understand such tasks easily, the issue of conflict resolution is not addressed. Despite these shortcomings, the heuristics may provide good starting points for rule-based modules. Imprecise rules such as "Traffic behind should be checked about every five seconds when there are vehicles also ahead"[40, 78] may be robustly captured by a fuzzy [32] formulation.

Motivational models [74] describe human cognitive activity during driving. By integrating perceptual information with mental states such as "perceived risk", these models predict the actions that a human driver would take in that situation. However since they do not describe *how* driving knowledge is represented, a large gap exists between the motivational model and its successful implementation. Specifying such models in a symbolic programming language [1] partially addresses this problem and SAPIENT may incorporate such risk estimates although not to this level of detail.

3.2. Accident Data Bases

Allen [4] surveys work in collision avoidance, including a number of accident data bases [30, 39] that identify the major types of crashes, along with their probable causes. Of these, SAPIENT will mainly be concerned with the following four: rear-end, lane change/merge, intersection/crossing path and single vehicle roadway departure. The scenarios described in Section 4.6 address many of these situations. Studies on lane changing/merging and intersection countermeasures [2] have also provided additional scenarios.

Reports from the Highway Research Board [7] and books on defensive driving [25, 79] present information that is important not only to human drivers, but also to designers of driving systems. The Traffic Engineering Handbook [48] compiles the vital parameters about lane widths and road curvatures that SAPIENT requires for its road geometry calculations.

3.3. Systems in Simulation

Task models have been used for reasoning by researchers at Linköping University[37, 44, 57, 61]. Each driving subtask is treated as an *independent* sequential procedure (car following, lane changing etc). A graph-oriented language designed to represent complex qualitative processes (Process Transition Networks) is used to determine which subtask should be executed in the current situation. However the assumption that the subtasks are independent is not always valid [8] and accounting for every possible interaction leads to extremely complicated networks.

Researchers at the University of Iowa have developed a driving system based on finite state automata [8, 13, 12]. After experiencing difficulties in designing and maintaining large single-layer automata, they are considering an implementation with multiple concurrent automata. The tradeoff in compiling driving knowledge into finite state machines is that the desired behaviors become hard-coded into the system. I believe that SAPIENT's proposed architecture will capture intentions in a more flexible manner.

Other researchers address driving as a distributed AI problem [43, 83] where several agents must solve conflicts in the absence of a central supervisor by cooperating without explicit communication. Hierarchical languages are used to model driving maneuvers at both a qualitative and quantitative level. Knowledge-based techniques reason at the former, while numerical methods simulate vehicle dynamics at the latter. Qualitative descriptions from sensors are used to extract actions of cars in the scene, in an attempt to infer their plans. However since perception is only considered at a very abstract level, with modules returning a "list of visible objects with accessible attributes"[83], it is unclear whether this approach is feasible given realistic sensor expectations.

Ulysses [52] is a system that drives effectively in its simulated road environment. Although perception modules are abstracted, a realistic cost is computed for each routine and Reece demonstrates that selective perception can reduce the total cost of sensing by several orders of magnitude. Although the overall strategy is sound, implementing Ulysses on a real vehicle is very difficult since the simulated sensors return more information about the world than is realistic (e.g. one can see around corners and through cars; sensors are error-free). Also, the perception system re-scans the entire region of interest during each cycle rather than tracking individual vehicles from frame to frame. Ulysses thus perceives its surroundings in a jerky manner — cars are remembered only as alarms, triggered to sound once the predicted path of the associated vehicle exceeds certain bounds. Hence there is no safeguard against cars that violate their expected behavior. Since our driver models are limited in their accuracy, SAPIENT should maintain a constant awareness of the immediate surroundings of the vehicle, and use intention-directed selective perception to obtain further information relating to higher-level goals.

A probability-based approach [45] (integrated with a rule-based guidance system) addresses the issue of uncertainty in sensing and decision making that is often ignored in simulation. By reacting to probable worst-case scenarios, a safe plan is generated for the given situation. This work addresses a limited scenario (driving on a straight road without intersections) and extending this method to the general driving problem may prove to be very difficult.

3.4. Systems on Vehicles

Although the majority of work on robot vehicles has been on the operational level, some lower level modules have addressed elements of higher level tasks. ALVINN [49] is a neural-network based road follower that outputs a confidence measure along with steering recommendations. This output confidence can detect potential single vehicle roadway departure cases [50]. SAPIENT shall use ALVINN as its road-follower and utilize the confidence measure in its reasoning.

RACCOON [63, 64] is a taillight tracking car follower. To correctly follow vehicles around curves, it possesses a basic understanding of situational awareness: it follows the *path* of the lead vehicle rather than heading directly for its current position. In RACCOON, the car tracking and path tracking aspects are decoupled, enabling the former module to be used by SAPIENT for vehicle information without modification.

DAMN [35] is an architecture where individual perception modules vote for and against a range of possible actions. The arbiter resolves conflicts and prioritizes the votes, selecting an action that best satisfies the given constraints. Although this approach has worked well in the cross-country domain, DAMN is unsuitable for driving in traffic since the lower level modules do not possess sufficient information about the situation to make informed votes (locally, each module sees multiple actions as equally feasible). The scenario presented in the Introduction is an example of such a situation.

4. Technical Approach

Since situational awareness incorporates many types of reasoning, I have chosen to implement it as a collection of *reasoning objects*. Objects will interact in a loosely coupled manner, with different objects operating at different speeds. The set of active objects will depend on the current situation and higher level goals. For example, a reasoning object involved in lane selection would only be activated before a planned lane change, while an object concerned with lane keeping would always be active. Plans generated by the slower objects may need to be pre-empted by important sensor information and may require real-time modification. Since the driving task prefers a good solution in real-time to an optimal one overnight, techniques that provide usable incomplete solutions are desirable.

4.1. Techniques for Traffic Reasoning

A number of techniques are potentially suitable for implementing situational awareness tasks. Although many of these approaches are computationally equivalent, some problems are more clearly expressed in one formulation than another. Rather than forcing all solutions to fit a single approach, SAPIENT's multiple object architecture will encourage designers to use the most appropriate technique for the task.

4.1.1. Rule-based Systems

Rule-based systems are already popular in the field. They can be implemented efficiently (or compiled into finite-state machines). A thorough protocol analysis can convert human expert knowledge into production rules, and much of the defensive driving literature already contains heuristics in the form of `if-then` cases.

To investigate rule based systems, a reasoning object was developed on the SAPIENT simulator to tackle an overtaking scenario on two-lane divided highways. The goal was to detect a slow-moving vehicle ahead, pass it and return to the original lane. Specialized sensors were mounted on the vehicles to sense cars in appropriate regions (ahead, rear-right etc). A 5-state automaton was able to solve the task from most initial conditions. By modifying the placement and parameters of the sensors, different behaviors could be induced into the system: for example, reducing the range of the rear-facing sensor created an aggressive system recklessly cut off the passed vehicle.

The primary problem with standard rule-based systems is that preconditions must match exactly, so the firing of a rule is often sensitive to ad hoc thresholds. Avoiding these "magic numbers" is usually impossible, but wherever possible, an attempt should be made to see whether they relate to driver preferences (e.g. desired speed, preferred car following distance). Also driver intentions are likely to become confused with situational constraints if one single-level network is constructed. Expert systems can capture context-dependent tradeoffs in rules but require a large knowledge acquisition effort [51].

A secondary problem is that these rules are very difficult to maintain and modify incrementally. A small local change can often have global repercussions, and a large number of new rules need to be constructed to handle the addition. This drawback can be avoided by constructing multiple concurrent systems, with an arbiter selecting the appropriate actions. It is not clear at this stage if a simple arbiter can perform this task correctly. Rule based systems that incorporate fuzzy or analytic reasoning address some of these concerns.

4.1.2. Fuzzy Logic

Fuzzy theory is based on the insight that categories are not absolutely clear cut [32] — for example a car does not become a "slow car" once its speed drops below some magic threshold; rather its "slowness" is a continuous variable that increases as its speed decreases. Linguistic variables allow an approximate match between the input and the antecedents of the rules [81] thus the problem of exact matching in traditional rule-based systems is avoided. This form of reasoning is very natural for people. For example, a rule for lane-changing may be stated as:

> Change lanes only if the target lane is free of fast cars behind you and slow cars ahead of you.

In fuzzy systems, knowledge is acquired symbolically, but represented numerically, and the traditional logical operators (e.g. AND, OR, NOT) acquire new semantics. The output is then defuzzified and mapped into actions.

Unfortunately, the fuzzification and defuzzification stages can involve arbitrary choices of functions, with a resulting loss of information. Furthermore, this approach is less useful in domains where we lack the introspection to generate the fuzzy if-then rules.

4.1.3. Analytic Models

Rule based and fuzzy rule architectures generally extract the heuristics for decision making from human experts. By using analytic models, we express the opinion that these heuristics need not be explicitly stated — rather that they will emerge in the system's behavior once we solve the general optimization problem of maneuvering our vehicle in traffic. An advantage is that our system's decision making is tailored to its sensory input rather than that of a human. By calculating acceleration constraints and predicting the potential trajectories of traffic around us over time, an optimization algorithm could present us with the optimal action for the given situation. Furthermore, this solution would have the advantage of a rigorous foundation.

Three analytical approaches were implemented in simulation: ribbons, acceleration constraints and gap reasoning. Each performed with partial success.

Ribbons are a representation of the local space around each vehicle. By examining other vehicles within this space, short-term predictions of their trajectories can be made. Although the approach is simple and efficient, it lacks the power to make longer term predictions. Furthermore, the sharp boundaries of the ribbons create a jerky behavior which is undesirable in highway driving.

Acceleration constraints are detailed in [52] where they are shown to generate acceptable behavior in many highway situations. Reece combines multiple constraints using their logical intersection, however he does not discuss cases where the logical intersection of goals is the null set (when all goals cannot be simultaneously satisfied). In such cases, the system should intelligently violate the less important constraints (such as posted speed limits) in order to satisfy vital goals (such as self-preservation). Without associated priorities, acceleration constraints are insufficient for decision making. Another problem is that acceleration constraint reasoning assumes that surrounding vehicles will begin braking at the same time as our vehicle. While this may be true for determining the safety of adjacent gaps, it breaks down for other gaps. In essence, acceleration constraints give us the correct answer to the wrong question. Reece comments:

> [Ulysses] does not search for a gap farther back in traffic if the adjacent one is not big enough. However, if the robot is blocked, traffic in the adjacent lane will tend to pass the robot, thereby providing new merging opportunities [52].

Unfortunately gaps in traffic tend to be dynamic, changing size over time, and pursuing a gap actively rather than waiting for the gap to become adjacent improves the chance of successfully merging into the desired lane. To address this, the third algorithm tracks gap characteristics (size, velocity, expansion rate) over time and selects the gap that optimizes an objective function as being the one most suitable for lane changing. Boundary conditions and lanes with fewer than two vehicles are handled with *virtual gaps*. Once a suitable gap has been found, the trajectory generator can be notified.

Analytic methods have a number of drawbacks which must be considered: the objective functions must be hand-crafted (a non-trivial problem in a complex domain); driver intentions are not easily captured (especially since other vehicles may react to occluded obstacles) and modifying large analytic systems with objective functions spanning different sub-systems is difficult. Lastly, such systems can be very sensitive to errors due to bad calibration.

4.1.4. Probabilistic Models

In driving, the behavior of vehicles around us is uncertain. This is partly because we do not have much information about each driver's goals, but also because other vehicles are affected differently by occlusions and road geometry. A probabilistic model is initialized with prior probabilities for a vehicle's behavior, and based on its observed actions, these probabilities are adjusted (possibly in a Bayesian manner). Given these probabilities, we calculate an expected utility function for our current situation, and choose the action that maximizes our utility. This approach can be combined with the optimization approach above, and its main advantage is that the uncertainty inherent in the traffic configuration is quantified in some precise manner. Also with this method, we can assess the risk of a given plan — and if this risk is acceptable (thresholds differ from driver to driver) then the plan is suitable.

This approach shares some of the drawbacks of the analytic models, but also has the added problem of deriving the prior probabilities for actions. Accident and traffic surveys give some statistics that may be of use, but many of the prior statistics are no more than ad hoc numbers. Furthermore, inferring the goals of a vehicle from observation is highly sensor dependent, and particularly if the car is not observed for very long, the probabilities are unlikely to differ substantially from the assigned prior probabilities.

Many of these prior probabilities cannot be assigned from known knowledge:

- What is the chance of a vehicle in the right lane taking an exit?
- What is the chance that there is a fast car behind that truck?

The elegance of this system is tarnished because we are forced to assign probabilities in ignorance of the other drivers' intentions. Although it is desirable to have some estimate of risk in a given situation, a strictly Bayesian approach may not be the right solution.

4.1.5. Hidden Markov Driver Models

Since probabilistic approaches that do not model driver intentions seem unrealistic, a possible solution is to model other drivers. Particularly in the case of a driver's associate, we may need to model our own driver (very few drivers will accept a system that requires a driver to explicitly state his/her intentions throughout the trip). If we can find a good model for drivers, then the probabilistic approach becomes attractive.

A hidden Markov model (HMM) [42] could be created for drivers, possibly with different parameters based on driver experience, aggressiveness and so on. Based on such models, it may be possible to hypothesize intentions, and given these intentions, predict the future behavior of the other vehicles. For example, if we observe a car approaching rapidly from behind us, in the left lane, and suddenly change into the right lane, where the traffic is slower, it may be reasonable to believe that the driver plans to take the next exit. With HMMs, a probability would be associated with this hypothesis, and this would be helpful for our risk assessment.

One big drawback of such modeling is that each vehicle around us must have such a model, and since many vehicles are within sensor range for only a short span of time, it is important to be able to initialize these HMMs to some reasonable default for each new vehicle. Whether or not we choose to model other vehicles, it may still be worthwhile modeling our own driver.

Hidden Markov models are also well suited for supervised learning, and the general comments about learning systems (See Section 4.1.6) also apply to learning HMMs.

4.1.6. Learning Systems

The many advantages of a learning system include: adapting to individual drivers, learning new system configurations and a possible improvement over hand-coded algorithms. However, the one requirement that both supervised and unsupervised learning systems have is training data. Let us consider each in turn.

Supervised learning (backprop neural-nets, HMM etc) requires training data and feedback from a teacher. In most cases, the learning phase is done off-line, though some systems, like ALVINN, have demonstrated impressive performance from on-the-fly training. Shepanski and Macy [59] developed a neural-network to control a car in an arcade game. Though the environment was greatly simplified, the system learnt to change speed and direction to stay on the road and avoid other cars. Interestingly, the system mimicked the driving style of its trainer: aggressive trainers created a system that drove recklessly while conservative drivers trained a network that would react passively. SAPIENT would benefit greatly from good driver models, and it may be possible to learn the preferences of our driver in a driver's associate application. It is unclear whether this can be used to model drivers of other vehicles.

Not needing teaching signals, unsupervised learning systems require a chance to explore their environment, with a way to assess their performance. In this domain, since exploring real highways is undesirable, a good simulator is needed. Unfortunately, designing a simulator that realistically models other drivers assumes that we already have good driver models. Therefore, I believe that unsupervised learning is more applicable to refining existing models. A good application of the learning approach would be a module that learnt the daily route, preferred speeds and car following headways of a commuting driver. More accurate models that know about the average lane deviation and corner cutting behavior of the current driver are also feasible.

4.1.7. Case Based Reasoning

Case based reasoning systems are well suited to ill-structured domains, where neither the system nor the human expert possesses causal domain knowledge [68]. In many applications (including tactical reasoning for driving), a user's optimization criteria are subjective and cannot be expressed in terms of a single global optimization function [54]. Expert systems have been criticized as being brittle, and CBR has the potential for dealing with noisy data [3]. The knowledge in a CBR system is implicit in the case base: the current situation is compared to situations in the case base, and an analogous solution is prescribed. If necessary, CBR systems can synthesize new solutions by combining snippets from solutions in the case base. And since these new situations can be added to the case base, CBR displays many of the benefits of a learning system. All of these factors indicate that CBR is a promising candidate for the tactical reasoning domain.

However, since previous experiences are used to guide problem solving, an effective similarity metric (used to find the best matches to the current situation) is critical. This similarity metric is generally hand crafted, and although causal knowledge is not required, the designer must still provide a suitable basis for matching. Furthermore, the solutions to a case must be represented in a form that allows synthesis.

A CBR implementation of the lane selection scenario is currently in progress. Hopefully tests in simulation, using real data will indicate if CBR systems are viable in this domain.

4.1.8. Hybrid Systems

Integrating the above techniques into hybrid reasoning objects is also possible. Examples include:

- A quickly cycling object with a reactive architecture implementing defensive driving heuristics, using fuzzy rules.
- A gap-finding loop triggered by lane-changing intentions utilizing a combination of analytic reasoning and probabilistic assessment of risk.
- A rule-based system for guiding selective perception at intersections, along with associated reasoning to determine if a left turn across traffic was possible.

4.2. Proposed Architecture

The different elements of SAPIENT's reasoning will be captured by individual *reasoning objects*, each of which may be implemented as a separate process. Objects will be created and destroyed based on the needs of the situation — for example, an intention to change lanes may spawn a gap-searching object, which would be destroyed once a suitable gap was selected.

An arbiter-like object will use the outputs of the reasoning objects to generate actions. This command fusion is necessary since different objects may recommend conflicting actions. The different objects need not operate in a synchronous manner — each may cycle at its own speed and buffer output for the arbiter. The arbiter object will cycle quickly and use the last output of a reasoning object (or some other reasonable default) in the absence of fresh output. This loosely coupled distributed architecture has the following advantages:

- No deadlock: the arbiter does not wait for the results of slow objects before recommending an action. In emergencies, it is essential to take any good action rather than waiting for the best action.
- Robustness: the failure of one object will generally not cause SAPIENT to fail. The malfunction of critical objects will compromise performance more than the loss of less vital objects.
- Incremental development: A new object could be tested individually and added to an existing system. Conflicting objects could be swapped out if needed.
- Modularity: Functionally identical objects (e.g. gap-finders) could be implemented in different ways. The proposed architecture will allow easy testing by enabling one to be swapped out for the other.
- Flexibility: SAPIENT could be used for either autonomous navigation or to assist human drivers by selecting the appropriate set of reasoning objects. For example, a driver's assistant may require objects that reason about the driver's intention.
- Parallelism: Different objects can be implemented on different processors or machines, increasing overall performance. Main consideration is the cost of shipping sensor data to different machines. Intelligent division of objects over machines can minimize this overhead.

An important difference between the internal representations of autonomous systems and semi-autonomous systems is that driver intentions need to be explicitly represented in the latter. In a driver's assistant system, a human driver may decide upon an action for reasons unknown to SAPIENT. For example, a hungry driver on the highway could spot a restaurant and decide to take the nearest exit. In the proposed architectures, *intentions* will trigger the creation and destruction of reasoning objects. In a driver's assistant system, the higher-level intentions would reflect driver intentions, while in a fully autonomous system, they would be generated by other objects in response to plans or current circumstances. Thus, intentions are the vocabulary for communication between reasoning objects.

SAPIENT will use existing modules for low-level perception tasks and higher-level strategic objectives. A possible architecture for this interaction is shown in Figure 2. The client module will provide suitable interfaces to human drivers or planning modules. Preferences and intentions are received from clients and recommended actions and warnings are returned. In the case of an autonomous system, the client module sends actions to the vehicle's controller.

4.3. Client Modules

I propose to develop two sample clients for use with SAPIENT: a driver's associate and an interface to the NAVLAB II controller. The first client will be a driver's associate with the following features:

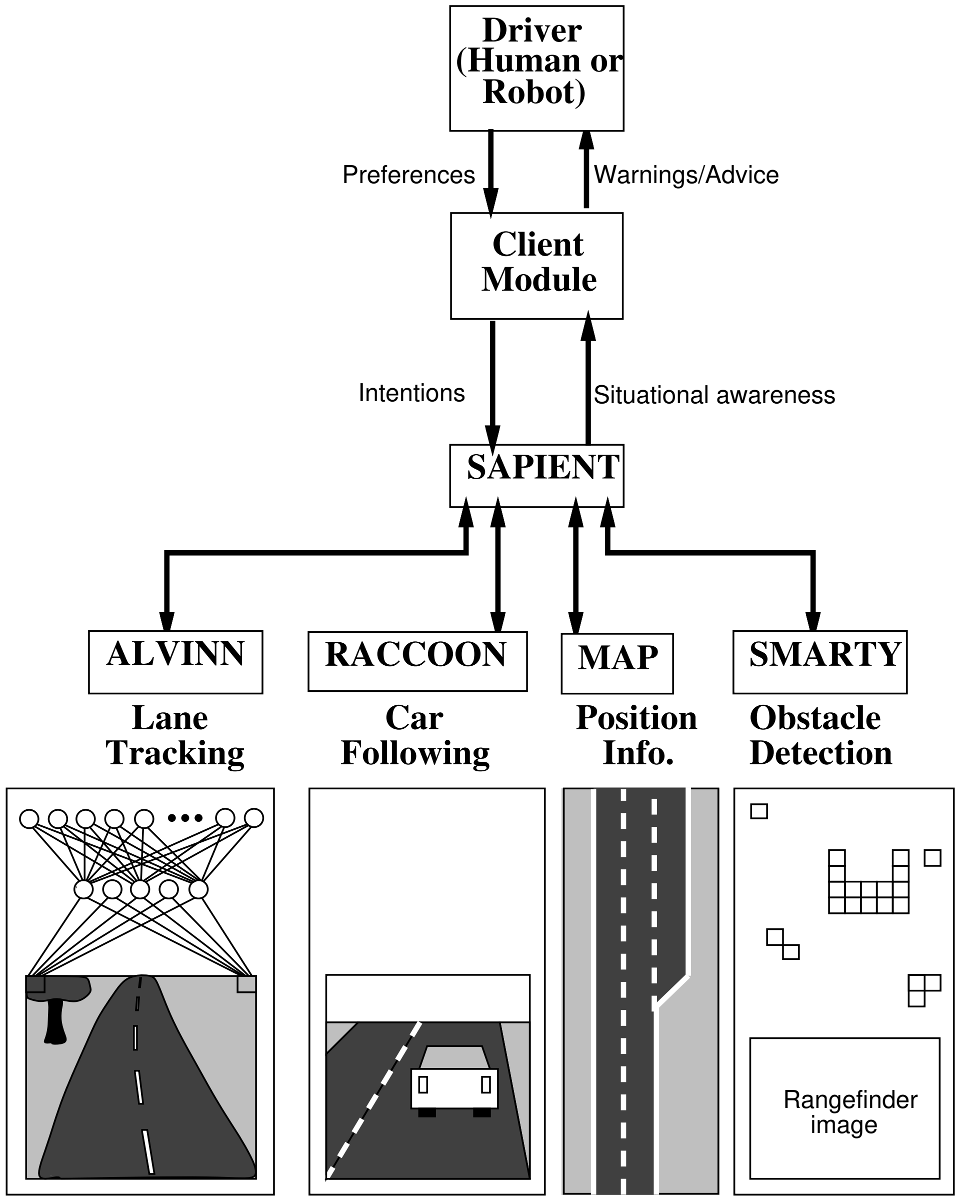

Figure 2: SAPIENT takes input from existing perception modules.

- Tailgating detection and warning.
- Warning driver when vehicle is in others' blindspots for extended period.
- A smart lane keeping monitor that understands lane changing.

Extensions of this client could take control of the vehicle in emergency situations or provide intelligent cruise control. The client should aid the driver without presenting irrelevant or dangerous information. Options must be provided early enough to be of use, yet not too far in advance [41]. Drivers who regularly travel a route are already aware of the traffic patterns on the road, and require a different level of assistance than drivers unfamiliar with their surroundings. Furthermore, the latter driver has less cognitive resources available to interpret drivers assistant output. Configuring the driver's associate for the needs of different drivers is an important issue.

The second client will be an interface to the NAVLAB II controller and allow limited autonomous navigation. Given suitable sensors, SAPIENT should be able to perform the simpler scenarios autonomously on the robot vehicle. For example, an autonomous overtaking system would check the surrounding traffic and safely pass slow lead vehicles.

4.4. Evaluation

SAPIENT's performance will be evaluated in both simulation and on real highways. The scenarios presented in Section 4.6 were extracted from real-life examples. I will construct reasoning objects for many of these scenarios and compare SAPIENT's solutions with those generated by other tactical reasoning systems (including experienced human drivers). Tests in simulation will enable different approaches to be compared quantitatively in the identical situations. Metrics which could be of use include:

1. Travel time (constrained by velocity limits).
2. Smoothness of path (minimizing lane changes, harsh braking).
3. Driving style (predictable behavior is desirable).

The scenario descriptions will be publicly available on the World Wide Web to invite solutions from developers of other tactical reasoning systems.

Limited testing on the NAVLAB II vehicle is also planned. Unlike many systems in simulation, SAPIENT has been designed with realistic sensor expectations. Since the simulation and robot environments will differ as little as possible, I hope to implement a version of SAPIENT on the vehicle which can solve some of the simpler scenarios. For safety purposes, tests in traffic will not be under computer control whenever possible.

4.5. Implementation

Although my goal is to drive SAPIENT in real traffic, highways are unsuitable testbeds in the initial stages of implementation because of safety concerns. Simulators, despite their imperfect modeling of the world, enable designers to develop their systems without risking lives and expensive equipment. By designing SAPIENT in simulation, I plan to:

- Repeatably test different algorithms in identical situations.
- Try new sensors and possible sensor failures.

- Recreate rare and dangerous traffic events.

Developing a system entirely in simulation also has its dangers: it is easy to make unrealistic expectations about sensor input; unmodeled vehicle dynamics may cause proposed actions to fail; and the behavior of real drivers may violate basic assumptions (e.g. the 2 second car following heuristic is rarely observed in real-life). For these reasons, I will also develop SAPIENT in parallel on a real vehicle. Although it may be infeasible to test the robot in all of the proposed scenarios, a demonstration of partial competence in real life is expected. I will ensure that objects developed in simulation will run with minimal modification on the real vehicle.

4.5.1. The Simulator

Ground vehicle simulators have been in use for many years for a variety of applications. Those which do not model vehicles individually (for example SIGOP [21]) are useful in traffic engineering, but too coarse for our needs. TEXAS [21], SIMRO [11] and NETSIM [80] are all microscopic but lack detailed modeling of lane changes and intersection traversal. VDANL [4] models vehicle dynamics and tire/road interactions accurately but has no provisions for multiple vehicle scenarios. Pharos [53] is a tactical-level simulator with individual vehicles which can calculate costs associated with simulated perception routines; unfortunately important perception difficulties such as occlusion, positioning errors and velocity estimation remain unmodeled. Alsim [27] is a simulator that focuses on sensing issues but is designed for unstructured environments, such as cross-country navigation. The University of Iowa simulator [8, 13, 12] incorporates realistic physical modeling of traffic situations along with an impressive user interface (including a real vehicle cab mounted on a flight simulator motion platform) but requires very specialized hardware and complicated scenario scripts. SAPIENT's simulator [65] is designed to address these shortcomings while running on a standard workstation in real-time.

4.5.2. The Robot

The robot testbed for SAPIENT is the NAVLAB II, an army ambulance with on-board power, sensors and computing. The sensors include video cameras, laser range-finders, radar and sonar. Positioning information is available from dead-reckoning, inertial navigation and GPS. A number of successful systems have operated the vehicle in both on-road and off-road conditions.

SAPIENT will use ALVINN [49], a neural-network based road follower for information about the current lane. Pre- and post-processing modules enable ALVINN to perform more than simple road-following. Virtual Cameras [23] creates an input image for ALVINN by rotating, translating and scaling pixels from a sub-image of the current camera image. An ALVINN network can be used to detect exits or adjacent lanes by selecting appropriate sub-images, and examining the confidence of the neural network. The Virtual Camera framework can also be used as a pre-processor by other vision-based perception modules to focus on regions of interest. Panacea [66, 67] uses an unmodified input from a steerable sensor to a single ALVINN network to correctly steer the vehicle by post-processing the output steering command.

Obstacle detection can be implemented using the ERIM, a laser range-finder with maximum range of 20 meters, 80 degree field of view and 2 Hz cycle time. Smarty [35] is a low-level perception module that discretizes the area around the vehicle into cells (0.4 meter squares), and based on the range-finder information, determines whether the cells are occupied or empty. My initial tests with real data have demonstrated the feasibility of using Smarty for car detection (with SAPIENT doing segmentation and tracking) however the short range, incomplete field of view and slow scan rate of the ERIM make it unsuitable for real highway applications. Stereo systems under development [56] may solve some of these problems.

Other modules on the NAVLAB which can interact with SAPIENT include: RACCOON [64] a vision-based car tracker; SAUSAGES [22], a toolkit for perception-based planning; and GANESHA [36], a grid-based local map module.

4.6. Analysis of Traffic Scenarios

In Section 2, scenarios were presented as a method of testing SAPIENT's competence.

Scenarios play two important roles in SAPIENT's development. Firstly scenario analysis indicates which reasoning objects are needed for situational awareness. Secondly, scenarios make good benchmarks for evaluating SAPIENT's performance. The scenarios in this section are classified into four levels of increasing difficulty: basic, intermediate, advanced and challenging. Scenarios from each of these categories are presented, along with observations on suitable approaches and desired sensors. Vehicle-based sensors refer to devices on our vehicle, while road-based sensors refer to stationary devices on the roadway. The latter are usually optional, and generally provide traffic or weather information.

4.6.1. Building Blocks

Road following and obstacle detection are often seen as prerequisites for driving systems. Similarly, for situational awareness, the following functions must be implemented before realistic scenarios can be tackled.

Mapping Obstacles into Lanes

A basic function of driving is associating obstacles with lanes. Humans use vision for both road-following and obstacle detection, and this task is performed unconsciously by the driver. However, if our vehicle is equipped with different sensors for each task (e.g. a camera for road-following and a laser range-finder for obstacle detection), this task must be explicitly performed by SAPIENT.

The simplest solution is to examine the bearing to each obstacle, and assuming a locally straight road, cast each obstacle to the appropriate lane. This approach is computationally efficient and works fairly well at low speeds with short-range sensors.

At highway speeds ($v = 30\, m/s$) however, longer range sensors are needed. For instance, to spot a stationary obstacle in time for emergency braking (assuming $a = -7 m/s^2$) a distance of 64.3 meters is needed. At this distance, the "straight-road" assumption breaks down — on a 12′ wide road, with radius of 500 meters, an obstacle at the minimum braking distance in an *adjacent* lane seems to be in *our* lane. False alarms from guard-rails on curved roads, and insufficient warnings about real obstructions in our lane are two obvious problems of such a simple method. A slightly more sophisticated idea is to use the local curvature of the road to place obstacles into lanes, but such an algorithm performs even worse on roads with S-shaped curves.

If a map of the road is given, and our positioning system is sufficiently accurate, the position of each obstacle can be converted to the map frame and off-road obstacles can be deleted. The positions of the remaining obstacles are best converted into longitudinal/lateral coordinates for further processing. The map of the lane structure may be obtained in advance, or provided by an appropriate perception module. The positioning system's output may be further enhanced by the road-follower's output, lane-trackers like Aurora [10] or from landmark recognition [70]. This method was implemented in simulation with excellent results, and initial tests using ERIM data from the Navlab II are also promising.

Speed Constraints

Speed constraints are a basic but important aspect of situational awareness. Road geometry, visibility, weather, vehicle performance, and posted signs are all factors which influence speed. The speed constraint due to road curvature is given by: $v = \sqrt{ar}$, where r is the curvature of the road, and a is the maximum allowable centripetal acceleration. For a road with $r = 500$ meters, and $a = 1m/s^2$ (a typical lateral acceleration at high speeds [40]), the velocity constraint is $22.4m/s$. This analysis assumed an unbanked segment; banked roads allow greater speeds.

The maximum allowable centripetal acceleration is lowered by wet weather and bad tire-road interactions. SAPIENT may be able to obtain weather information from on-board sensors and road-type information from the map, and recommend lower speeds in non-optimal conditions. Since these factors also affect braking distance, they will also constrain speeds on straight roads.

Blindspot Reasoning

Most vehicles have gaps in their field of view known as blindspots. Unless the driver consciously turns his/her head, vehicles in these areas are not visible (despite rear-view mirrors). In the case of vehicles such as trucks or buses, these areas can be fairly large. The following advice is often displayed on the backs of trucks: “If you can’t see my mirrors, I can’t see you.” Experienced human drivers are careful not to drive in the blindspots of neighboring vehicles for long periods of time.

Blindspot reasoning consists of two main components:

1. Monitoring vehicles in our blindspots.
2. Avoiding driving in other vehicles’ blindspots.

Both components require a robust car tracker working with the on-board sensors. Omni-directional perception is strongly recommended for our vehicle and one possible way to achieve this is to mount two spinning 1-D “light-house” range sensors at opposite corners of the vehicle.

Geometric reasoning is required to determine whether a vehicle is in our blindspot, and whether we are in another vehicle’s blindspot. The former case is relatively straightforward since our vehicle’s blind areas are known; the latter case may require hypothesizing the class of the other vehicle (e.g. car, truck, bus, motorcycle) and reasoning based on some prototypical model of the vehicle (i.e. trucks have a blind area immediately to their rear).

A blindspot reasoning module was successfully implemented in simulation. A counter was incremented when our vehicle was in another vehicle’s blindspot and when the counter exceeded a threshold, a warning was generated. The disadvantage of this simple algorithm is that the warning is independent of the danger of the traffic configuration.

A more sophisticated analysis based on the likelihoods of vehicles being adversely affected by cars in their blind areas could give an overall risk factor for the configuration. Interpreting this risk factor is more natural than using the ad hoc time thresholds discussed above.

4.6.2. Basic

The basic scenarios require minimal sensing and very little long-term planning. Most objects that implement these aspects of situational awareness need to be active all the time. In general, the objects will also have fast cycle times (a reflection of the simpler reasoning involved).

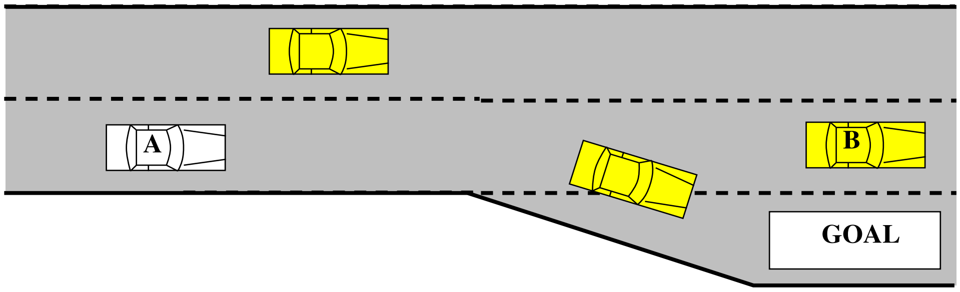

Figure 3: Scenario: Leaving the Highway

Preventing Roadway Departures

Run off road incidents account for 21% of all accidents and 37% of all fatal accidents in the U.S. [2]. It is important for a driver's associate system to recognize such dangerous situations without generating false warnings during overtaking or lane changing maneuvers. This scenario is less applicable in the autonomous navigation domain, since a competent road follower module is assumed to be active at all times.

In the simplest implementation, the output from a road following module could be compared with the steering response of the human driver and if the difference exceeded a certain threshold, an alarm could be sounded [50]. The module could be manually suppressed during lane changing maneuvers. Unfortunately, this approach has two obvious drawbacks:

1. Driver intervention is necessary to temporarily suppress the system.
2. If a roadway departure occurs during a lane change maneuver, the system is powerless to detect or correct it.

To address the first problem, we need to obtain information about the intentions of the driver. A straightforward solution would be to monitor the state of the vehicle's turn indicators (though this scheme is not foolproof) and allow lane changing only while the appropriate indicator was activated.

To solve the second problem, we could compare the trajectory of the vehicle with a lane changing model. If the trajectory deviated substantially from the predictions, we would assume that the driver had lost control of the vehicle. Weiss's characterization [77] of a lane change as an overlay of straight lines and a sine curve may be suitable.

Unfortunately, different drivers' styles would match this model to different degrees. An enhancement would be to learn typical values for the model parameters for the given driver during a series of test drives. A customized drivers' associate of this type would be able to detect roadway departures more reliably without generating as many false alarms. Furthermore, since ALVINN (the road follower) can be trained by the specific driver, reliable performance during lane keeping may also be expected.

Leaving the Highway — Taking an Exit

The problem of leaving the highway (assuming that our vehicle is already in the correct lane) is easier than the general problem of lane changing or merging, since the target lane is unoccupied at the time of our arrival (See Figure 3).

Since the areas of interest are few and clearly defined, it is reasonable to employ analytic methods to calculate the relevant acceleration constraints. Also unlike most lane change situations, timing is not an issue — we can change lanes as soon as we are adjacent to the free lane.

The exit ramp generates speed constraints (usually from road geometry and posted speed limits), and may have a lead vehicle, which imposes a car following constraint.

The sensor requirements for this scenario are fairly modest, making it suitable for implementation on the Navlab II: a forward facing sensor for vehicle detection, and a good road-following module. Virtual Cameras (in conjunction with ALVINN) can be used to detect when the exit lane comes adjacent.

Note that some highways are designed so that the right-most lane leads directly into the exit ramp, eliminating the need for a lane change — in these cases, only the car following constraints need to be considered.

Maintaining a Space Cushion

Human drivers are taught to keep a space cushion around their vehicle [58]. This heuristic has two main benefits:

1. Increased visibility: occluded obstacles that appear from behind other vehicles are spotted at a greater distance. This allows more time for decision making.
2. More room for maneuvers: sudden lane changes are possible only if a suitable gap exists adjacent to our vehicle; an emergency braking maneuver is dangerous if we are being tailgated.

Most lane changing accidents occurred because the drivers did not see the other vehicles [29]. One reason for this is blindspots, which were discussed above; the other main reason is occlusion — many of the more challenging scenarios involve occluded vehicles taking unexpected actions. Drivers who maintain a space cushion are less likely to find themselves in such no-win scenarios.

Many approaches may be useful for this scenario: rule-based, fuzzy, analytic and case-based all look promising. Some situations can be recognized and corrected — for example one may respond to tailgating by speeding up or changing lanes. Sensors that can detect the positions and velocities of cars around us are essential for this scenario.

4.6.3. Intermediate

The intermediate scenarios require more reasoning, and most involve some degree of look-ahead. Human drivers plan effectively without exact knowledge of how other drivers will act; experienced drivers realize that these plans may need to be aborted if events do not transpire as predicted.

Lane Changing

Deciding if a particular lane change is safe or not can be a non-trivial tactical level problem. Certain assumptions must be made about the intentions of other drivers in the area, and perceptual difficulties such as occlusion need to be considered. This scenario examines some of the problems that can occur during lane changing.

In Figure 4, the shaded vehicles (B, C) represent real vehicles around the robot while the light vehicles with dashed lines (P, Q, R) are some areas where hypothetical vehicles need to be considered. Potential problems with the proposed lane change include:

1. Ramming Car B ahead of us, due to excessive acceleration.

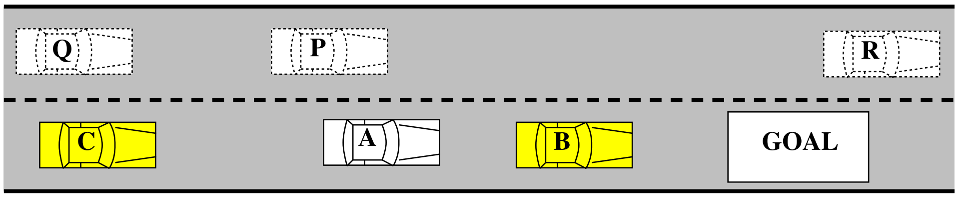

Figure 4: Scenario: Lane changing

2. Abrupt deceleration (to get behind a Car P), inviting a rear-end collision with a car C behind us. This is especially dangerous if C is tailgating us prior to the lane change.
3. Sideswiping car (P) beside us (particularly likely if our vehicle has a blind area to the side).
4. Changing lanes in front of a rapidly moving car (Q) behind us in the target lane. In general this will not cause a collision unless Q is near us, though Q's driver will see this as an impolite maneuver.
5. Prematurely changing lanes behind a slow vehicle (R) in the target lane. Though this may not lead to an accident, such a maneuver is not efficient — it is better to let R fall behind us before changing lanes. In certain cases the action may be justified (e.g. congested roadway when we need to take a left-lane exit).

To ensure a safe lane change, each of these conditions should be checked. In particular, the hypothetical cars (P, Q, R), which cannot always be seen from our current position, (due to occlusions) should be expected. Human drivers often change position slightly within their lane, both laterally and longitudinally, to see these cars before committing to a lane change. Also a lane change should be aborted if one of these conditions seems very likely.

Depending on the speeds of the gaps in the target lanes, our vehicle may need to accelerate or decelerate in the current lane. A strategy employed by humans when passing [69] is to temporarily lag behind the lead vehicle to create space for this acceleration. Such actions require at least some amount of short term planning.

The lane changing problem can be addressed successfully by a rules based system, since the important considerations are concisely enumerable. The passing scenario was successfully implemented in simulation. Both fuzzy logic and analytic reasoning are feasible approaches for capturing the relations between distance and velocity for the vehicles in the passing lane. Good driver models and learning our driver's intentions will make our solutions more robust but neither are required elements for this scenario.

Road-based sensors providing traffic density information can help in the lane selection process, but are not needed during the actual lane change maneuver. Sensors for determining the positions and velocities of surrounding vehicles are critical. It is highly recommended that such sensors be omni-directional since complicated lane changes may require perception in all directions.

Lane selection is considered in greater detail in Section 4.6.5

Entering the Highway — Merging into a Traffic Stream

Figure 5 depicts a typical merge scenario. The acceleration lane (on well designed highways) enables our vehicle to match speeds with the traffic stream in the target lane. The driver's task is to select a suitable gap in the target lane, and change lanes before the acceleration lane ends.

Since most of our attention is focused on cars in the target lane, accidents where Car C is rear-ended by our vehicle are quite common. SAPIENT should recommend leaving sufficient maneuvering space to allow changes in speed and position without interference from Car C.

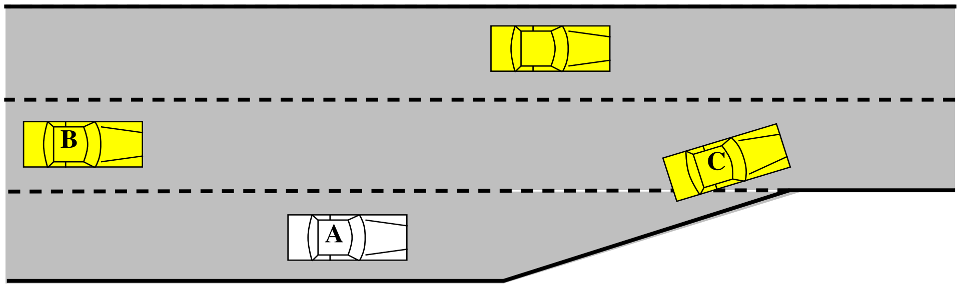

Figure 5: Scenario: Entering the Highway

Merging is a fairly passive operation since gaps in the target lane must present themselves (in most conditions, suitable gaps exist since cars in the target lane have changed left into a faster lane to avoid interacting with merging vehicles). Gap analysis can be used on the spaces in the target lane, to find a suitable gap. Confident drivers are likely to accept smaller gaps with less room for error. In congested situations, it may become necessary to accept gaps which temporarily result in an unsafe roadway, forcing trailing vehicles to slow down in order to avoid a collision. Experienced drivers are able to judge the expansion rates of gaps accurately, and in such cases, the timing of the lane change is critical.

In merging scenarios, the most important factor is the traffic density in the target lane. We define D, the average size of a gap, to be:

$$D = v/k \tag{1}$$

where v is the average velocity of traffic in the target lane, and k is the number of cars that pass by in one second. If D is larger than the acceptable gap size, a merge should be relatively easy; conversely, if D is smaller than the acceptable gap size, a longer wait may be anticipated. Road-based sensors which can give information about traffic flow rates can make the merging problem easier. In either case, vehicle-based sensors with good velocity estimation are critical. Merging should not be attempted without a rear-facing sensor.

Some entrance ramps have STOP signs posted at their end. Such ramps require an accelerative lane change — we need to accelerate to the speed of car B before B enters the unsafe car following zone. This can be difficult if the distance to vehicle C is very short. Note also that the speed that must be achieved in the target lane is determined by B rather than the general speed of the lane, so merging in front of a fast car is not recommended.

4.6.4. Advanced

Four-Way Intersection

Intersections require a high degree of situational awareness because our vehicle can simultaneously interact with a number of lanes of traffic. Intersections without traffic signals are especially challenging. Figure 6 depicts a typical four-way intersection on a multi-lane highway. Our car is in the left lane, intending to perform a left turn from a two-lane road onto a divided highway, with two lanes going in each direction, across traffic. In the absence of a traffic light, cars A and D usually face STOP signs.

Possible interactions can occur with the following vehicles:

- Car C′, in the target lane, behind the goal gap.
- Cars B and B′, headed in the opposite direction may arrive in the intersection while our vehicle is crossing.

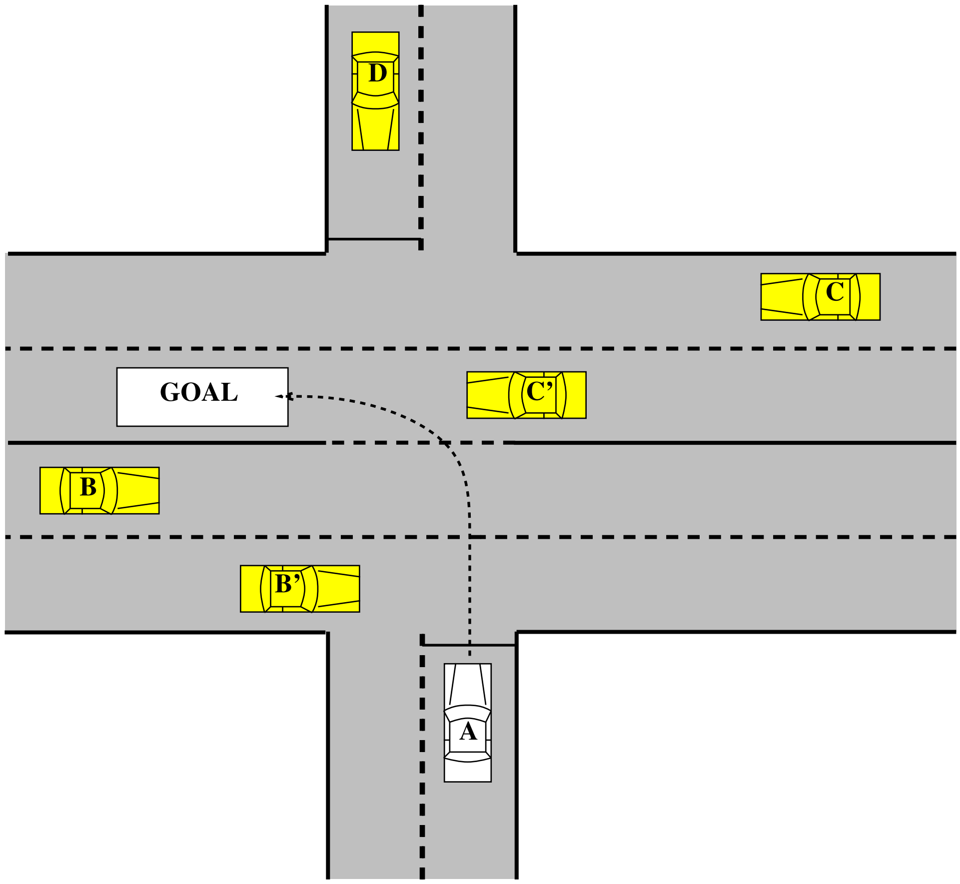

Figure 6: Scenario: Four-way intersection

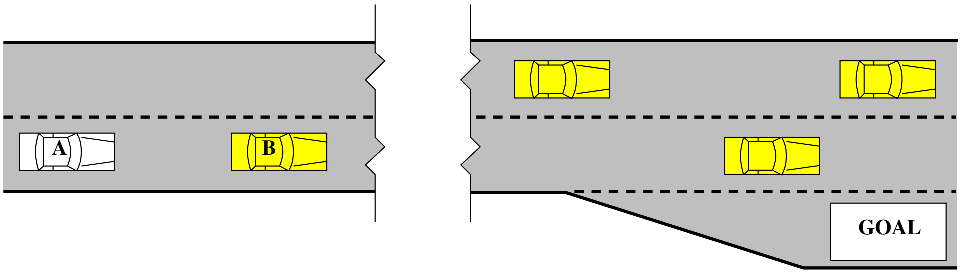

Figure 7: Scenario: Advanced lane changing

- Car D, currently stationary, may be planning to enter the intersection.

Of course, if this intersection has a traffic signal, only car D needs to be considered (assuming that the other vehicles obey the signal). It is also important to assume that a visible vehicle (especially if large, like a truck) may partially or fully occlude other vehicles that may need to be considered. The general strategy is to verify that none of the paths that cross our intended path have vehicles within range.

For some intersections, particularly those with STOP signs, protocols for deadlock resolution are needed. Rule-based systems may be suitable since such heuristics are well known. Analytical reasoning can provide information about gaps at uncontrolled intersections with light traffic (where we want to make a turn through oncoming traffic). Though a case based system may be well suited to making the lane selection decision at the intersections, CBR is not needed to negotiate the intersection itself.

Road-based sensors that can give positions and velocities of vehicles at intersections will be very useful. In their absence, we need vehicle-based sensors with a wide field of view (preferably omni-directional). Our sensors must examine cars coming from a variety of directions, and selective perception modules that can answer queries about cars on different roads will be useful.

Advanced Lane Changing

Figure 7 depicts the scenario described in the Introduction. Our vehicle is in the right lane of a divided highway, approaching the chosen exit. A slow vehicle B blocks the right lane, but the traffic in the left lane is moving at our desired speed. In some circumstances, lane-markings may prevent lane changing before the exit (e.g. the road may go through a tunnel). Our desire to pass Car B conflicts with our desire to successfully take the exit and none of the lower level modules (road follower, car follower or lane changer) have sufficient information about the scenario to select the appropriate action on their own. The decision to change lanes must come from our tactical reasoning of the situation, based on a combination of map information, current vehicle configurations and any data we may have of the traffic conditions at the exit.

Human drivers familiar with the road make such decisions routinely, since they have developed an understanding of the prevalent traffic conditions near the exit, and know the distance to the exit from their current position. Some of this knowledge is captured on the map, and a road-based sensor near the exit may give us traffic density information. If the average gap size (from Equation 1) is small, then making the pass could result in missing the exit.

Different approaches could solve this scenario, and it is unclear which one is best at this stage. Given a domain expert, a rule-based system may be successful, although drivers often make these decisions based on a hunch. This gestalt evaluation of the current state is captured well by case-based reasoning, provided that sufficient data about such situations can be collected. Calculating the risks involved (chance of finding

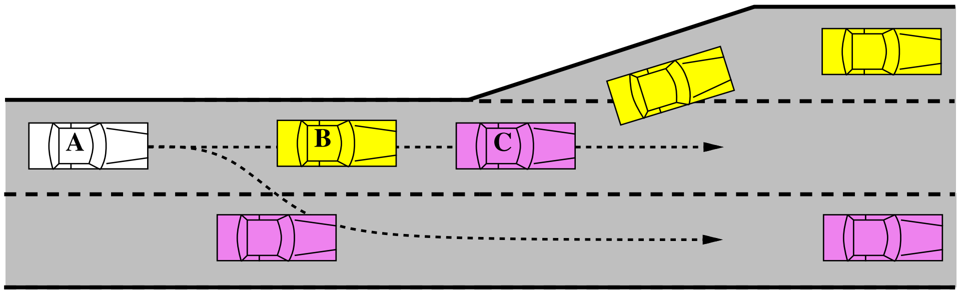

Figure 8: Scenario: Lane Selection

a suitable gap near the exit etc.) could prove to be valuable — for example a display could show the approximate risk of missing the exit, along with a confidence value. Aggressive drivers could choose to take the gamble, while more conservative drivers would stay in their right lane knowing that an acceptable solution was guaranteed by their current state. A learning system could observe the driver responses based on the risk factor and learn a particular driver's preferences. Such a system may be able to suggest good lane changes that the driver would otherwise have missed.

Many of these systems assume some knowledge of driver intention. A system that assumes every exit is the one you want to take clearly hinders more than it helps. If the route was known in advance (most commuters take the same route to work and back daily), then the system would already know which exit to monitor. Another alternative is to have the driver query the system (maybe by voice): "Computer: I'd like to pass the truck in front of me and take exit 8. Can I do it before we get to the tunnel?".

In the absence of road-based sensors by the exit, general knowledge about traffic conditions in the area would be very useful (e.g. Exit 13 is crowded during morning rush hour, but otherwise fairly empty). Vehicle-based sensors that can judge positions and velocities accurately are indispensable.

4.6.5. Challenging

Lane Selection

Figure 8 shows a situation where the length of the left turn lane is critical in determining traffic flow. When the number of vehicles that want to turn left at this intersection exceeds the capacity of the left turn bay, cars begin to block the center lane. Although the average speed of the center lane is generally higher at such intersections, the left-turning cars (which need more time) may make the right lane preferable in such circumstances. Assuming that we want to go straight, which lane should we choose?

Although both choices lead to the same goal (going straight at the intersection), one of the choices is clearly better in terms of elapsed time. Selecting the better lane is a difficult problem requiring knowledge about:

- Road geometry: which lanes lead to our destination?
- Traffic signals: e.g is there a special left turn signal here? How long does the light stay green?
- Intentions: e.g. how many cars are planning to turn left?
- Traffic patterns through the intersection: average speeds of each lane.

Some of this information can be provided by road-based sensors (e.g. number of cars through each lane per unit time). Vehicle-based sensors need to cover all potential lanes, though velocity information is not as critical in this scenario. Advance warning of the intersection (from a map) is desirable, so that the lane

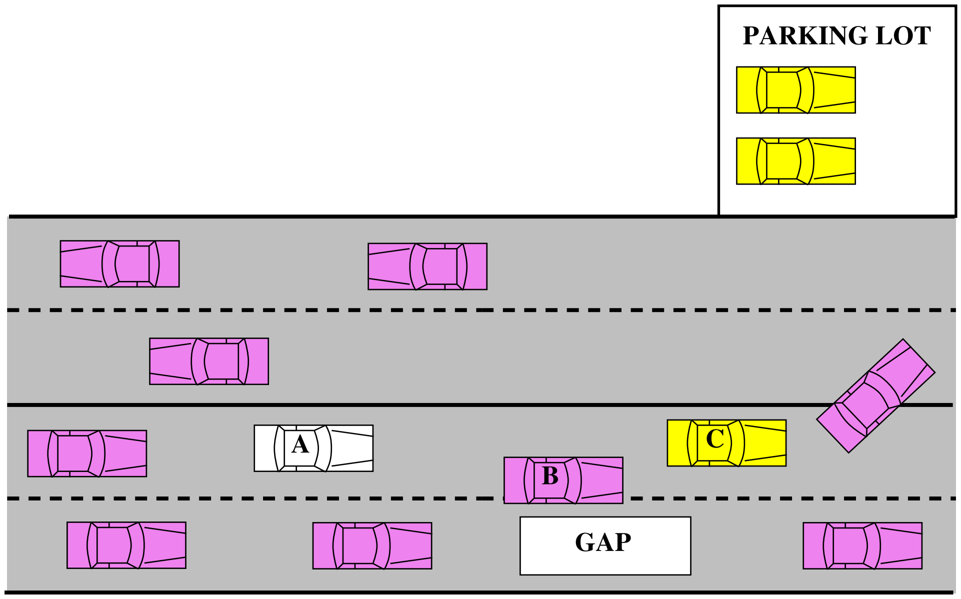

Figure 9: Scenario: Anticipating danger

selection can be made while gaps still exist in adjacent lanes. During peak hours, many intersections do not clear completely during the green light phase, and there is a backlog of vehicles in all lanes. These situations are simpler to analyze since we only need to know about the cars at the rear of each queue.

Finding good heuristics for the lane selection problem is difficult because of the conflicting factors involved. Case based reasoning may be able to capture the salient features in a more robust manner. Assuming that data is collected on the intersection under different conditions (different times of day, with different numbers of vehicles etc), it seems likely that a good case base could span the possible situations fairly well. A CBR implementation for this scenario is currently in progress.

Anticipating Danger

Some scenarios are unsolvable by the time they are recognized and must therefore be avoided rather than solved. Human drivers find these situations challenging, and those who travel such routes must learn to anticipate these situations [69]. Figure 9 shows an undivided four-lane highway with businesses on both sides. The average speed of both lanes is high, yet cars in the left lane may be stopped, waiting to turn left across the double yellow line.

Our vehicle, car A, is following car B with sufficient headway for highway driving, at high speed in the left lane when car B spots stationary car C. Although the distance between cars B and C is smaller than car B's braking distance, an adjacent gap makes lane changing feasible. From our perspective, car C is occluded so we are not aware of any danger until car C changes lanes. Unfortunately, by now we are too close to car C to brake in time. Unless there is a suitable gap for us in the right lane, a collision is inevitable. And since vehicles behind us are also unaware of the threat, a multiple-car collision is extremely likely.

The problem with this scenario arises from a violation in an assumption in our car following heuristic — the follower assumes that the lead vehicle will brake in time to avoid a collision with vehicles ahead. Vehicles behind the leader do not keep a headway equal to braking distance — only sufficient headway to account for their reaction time. The lead vehicle in this scenario avoided a collision by finding a gap, however this does not guarantee that any the followers will be able to do the same. Although the danger was present *before* the lane change, we were unable to perceive it. Good drivers are aware of such potential dangers.

Defensive driving plays an essential role in solving this scenario.

Static analysis on the map information should reveal areas where this scenario is likely (e.g. undivided highways with buildings on the other side). Traffic sensors should also indicate medium to heavy traffic since the danger exists only when there are closely spaced vehicles. Once SAPIENT has determined that this scenario is likely, the system should recommend two precautionary measures:

1. Slow down (to decrease braking distance).
2. Increase headway to lead vehicle (increase warning time).

SAPIENT should also recommend keeping a space cushion (see earlier scenario) for emergency lane changing.

Given these defensive precautions, even no-win situations like this can be successfully negotiated. The sensors required for the scenario include vehicle detection (including side- and rear-facing ones for lane changing); map based information; and traffic sensors to give average gap size or information about stopped vehicles (if possible).

5. Research Agenda

SAPIENT's development process will involve the following elements:

1. Simulator refinement.
2. Scenario analysis.
3. Reasoning object implementation.
4. Testing in simulation.
5. Testing on the NAVLAB II.
6. Evaluation.

The simulation environment must reflect the desired robot environment closely. In particular, abstracted perception should be avoided. Currently, scanning laser range-finders have been implemented, and I propose to implement other sensors as they become available on the NAVLAB II.

Scenario analysis motivates algorithms in reasoning objects. The scenarios described in this document are a representative sample of situations rather than a canonical listing. I will continue to look for other interesting scenarios, mainly for evaluation.

Reasoning objects form the core of the intelligence inside SAPIENT. I plan to focus on implementing these objects in an efficient manner (suitable for real-time use). Since SAPIENT's architecture encourages incremental development, I propose to create the objects for basic reasoning first and test them in parallel with implementation.

Initial testing will occur in simulation, mainly for safety reasons. I hope to test subsets of SAPIENT on the NAVLAB II early since the simulation environment matches that of the robot very closely. Although the NAVLAB II currently lacks suitable vehicle sensors, the situation is expected to improve by the time I am ready to test SAPIENT.

A qualitative evaluation of SAPIENT will be provided by examining its performance on the scenarios in simulation and on the road in real-life. A quantitative measure based on metrics such as minimized time or smooth path is also feasible. I hope to make comparisons with other tactical reasoning systems and human drivers; for the latter, a realistic interface is desirable. I will also examine the improvement in SAPIENT's performance as better reasoning objects are implemented.

6. Expected Contributions

My goal is to develop a tactical reasoning system capable of making driving decisions in real-time. I believe that situational awareness lies at the heart of any such system, and that qualitatively different tasks need to be performed. I will refine existing approaches and formulate new techniques for navigation in traffic, and implement them as *reasoning objects* in a robust, real-time architecture.

SAPIENT will form the core of a new intelligent vehicle system, with potential applications including drivers' assistant systems and tactical reasoning modules for autonomous navigation in traffic. The approach will be applicable to many tasks involving the integration of noisy and incomplete sensor data with higher level objectives.

7. Schedule

Fall 1993	Develop simulator for SAPIENT Collect scenarios. Perform simple experiments on simulator.
Spring 1994	Implement analytic techniques. Collect and analyze scenarios. ERIM/Smarty experiments on NAVLAB II.
Summer 1994	Test some approaches on simulator. Investigate architectures/approaches. Write thesis proposal.
Fall 1994	Thesis proposal presentation. Design multiple object architecture. Implement objects for basic scenarios. Examine some difficult scenarios.
Spring 1995	Test basic SAPIENT in simulation. Implement a difficult scenario. Integrate basic SAPIENT into NAVLAB.
Summer 1995	Implement a Driver's Associate client. Test difficult scenarios in simulation. Test subset of SAPIENT on NAVLAB.
Fall 1995	Extensive testing in simulation. Test SAPIENT Driver's Associate on Navlab II. Begin writing thesis.
Spring 1996	Finish writing thesis. Defend.

References

[1] Jans Aasman. Implementations of car-driver behaviour and psychological models. In J. A. Rothengatter and de Bruin R. A., editors, *Road User Behavior: Theory and Practice*. Van Gorcum, Assen, 1988.

[2] Federal Highway Administration. Potential safety applications of advanced technology. Technical Report FHWA-RD-93-080, U.S. Department of Transportation, Washington D.C., U.S.A., January 1994.

[3] D. W. Aha. Case-Based learning algorithms. In *Proceedings of the Case-Based Reasoning Workshop*, pages 147–159, 1991.

[4] R. W. Allen. The driver's role in collision avoidance systems, 1994. Paper prepared for presentation at the Workshop on Collision Avoidance Systems.

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

[6] J. G. Bender and R. E. Fenton. A study of automatic car following. *IEEE Transactions on Vehicular Technology*, 18:124–140, November 1969.

[7] Highway Research Board. Accident analysis and speed characteristics. Technical Report Bulletin 341, Highway Research Board, Washington D.C., U.S.A., 1991.

[8] 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.

[9] 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.

[10] Mei Chen, 1994. Personal Communication.

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

[12] J. Cremer and J. Kearney. Scenario control for virtual environments. In *Proceedings of the IMAGE VII Conference*, June 1994.

[13] J. Cremer, J. Kearney, Y. Papelis, and R. Romano. The software architecture for scenario control in the Iowa driving simulator. In *Proceedings of the 4th Computer Generated Forces and Behavioral Representation*, May 1994.

[14] J. W. Cro and R. H. Parker. Automatic headway control — an automobile vehicle spacing system. Technical Report 700086, Society of Automotive Engineers, January 1970.

[15] 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.

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

[17] 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.

[18] K. Fujita et al. A knowledge-based system for autonomous highway cruising. In *Proceedings of the Japan-USA Symposium on Flexible Automation*, 1990.

[19] D. W. Gage and B. J. Pletta. Ground vehicle convoying. *SPIE Mobile Robots II*, 852:319–328, 1987.

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

[21] 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.

[22] Jay Gowdy. Object oriented architecture and planning for outdoor mobile robots, 1994. Thesis Proposal.

[23] Todd M. Jochem. Using virtual active vision tools to improve autonomous driving tasks, 1993. Thesis Proposal.

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

[25] Paul W. Kearney, editor. *How to Drive Better and Avoid Accidents*. Thomas Y. Crowell Company, New York, 1963.

[26] N. Kehternavaz, N. Griswold, and S. Lee. Visual control of an autonomous vehicle (BART) — the vehicle following problem. *IEEE Transactions on Vehicular Technology*, 40(3), 1991.

[27] Alonzo Kelly. A rudimentary mobile robot simulator. Carnegie Mellon University, 1992.

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

[29] R. R. Knipling. IVHS technologies applied to collision avoidance perspectives on six target crash types and countermeasures. In *Proceedings of the IVHS AMERICA Annual Meeting*, April 1993.

[30] R. R. Knipling, D. L. Hendricks, et al. A front-end analysis of rear-end crashes. In *Proceedings of the IVHS AMERICA Annual Meeting*, May 1992.

[31] R. Kories, N. Rehfeld, and G. Zimmermann. Towards autonomous convoy driving: Recognizing the starting vehicle in front. In *Proceedings of the 9th International Conference on Pattern Recognition*, 1988.

[32] Bart Kosko. *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence*. Prentice-Hall, 1991.

[33] D. Kuan, G. Phipps, and A. C. Hsueh. Autonomous robotic vehicle road following. *IEEE Transactions on PAMI*, 10(5):648–658, 1988.

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

[35] D. Langer, J. K. Rosenblatt, and M. Hebert. A reactive system for off-road navigation. In *Proceedings of IEEE Conference on Robotics and Automation*, 1994.

[36] Dirk Langer and Charles E. Thorpe. Sonar based outdoor navigation and collision avoidance. In *Proceedings of IROS*, 1992.

[37] J. Malec. How to pass an intersection, or automata theory is still useful. Technical Report LiTH-IDA-R-91-08, Department of Computer and Information Science, Linkoping University, Linkoping, Sweden, 1991.

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

[39] D. L. Massie, K. L. Campbell, et al. Development of a collision typology for evaluation of collision avoidance strategies. In *Thirty-Fifth Annual Proceedings, Association for the Advancement of Automotive Medicine*, October 1991.

[40] J. McKnight and B. Adams. Driver education and task analysis volume 1: Task descriptions. Technical report, Department of Transportation, National Highway Safety Bureau, November 1970.

[41] Richard M. Michaels. Operational issues in driving performance using route guidance systems. *IVHS Review*, Winter/Spring:35–42, 1994.

[42] 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.

[43] R. Mock-Hecker and M. Zeller. Recognizing hazardous situations among vehicles in a traffic world. In *Proceedings of the 4th Annual Conference on AI, Simulation and Planning in High Autonomy Systems*, September 1993.

[44] Simin Nadjm-Tehrani. Analysis of the overtaking scenario: Specification of an autonomous car and a driver support system. Technical Report LiTH-IDA-R-91-07, Department of Computer and Information Science, Linkoping University, Linkoping, Sweden, 1991.

[45] Alex Niehaus and Robert F. Stengel. Probability-based decision making for automated highway driving. *IEEE Transactions on Vehicular Technology*, 43(3):626–634, August 1994.

[46] Okuno, Fujita, and Kutami. Visual navigation of autonomous on-road vehicles. In Ichiro Masaki, editor, *Vision-Based Vehicle Guidance*. Springer-Verlag, 1992.

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

[48] James M. Pline, editor. *Traffic Engineering Handbook — 4th Edition*. Prentice-Hall, 1992.

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

[50] Dean A. Pomerleau, 1994. Personal communication.

[51] D. S. Prerau. *Developing and Managing Expert Systems*. Addison-Wesley, Reading, MA, 1990.

[52] Douglas. A. Reece. *Selective Perception for Robot Driving*. PhD thesis, Carnegie Mellon University, May 1992.

[53] 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.

[54] C. R. Reeves, editor. *Modern Heuristic Techniques for Combinatorial Problems*. Halsted Press, New York, 1993.

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

[56] Bill Ross. A practical stereo vision system. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1993.

[57] E. Sandewall. Proposal for a ProART specification platform. Technical Report LAIC-IDA-90-TR18, Department of Computer and Information Science, Linkoping University, Linkoping, Sweden, 1990.

[58] Motor Vehicle Services. *New Jersey Driver Manual*. New Jersey Department of Law and Public Safety, 1988.

[59] J. F. Shepanski and S. A. Macy. Manual training techniques of autonomous systems based on artificial neural networks. In *Proceedings of IEEE IJCNN*, 1987.

[60] Texas Engineering Experiment Station. BART: Binocular Autonomous Research Team. Technical report, Texas A&M University, 1989. Research Brochure.

[61] Frank J. J. M. Steyvers, editor. *Traffic Research Center: Annual Report 1989*. CIP-gegevens Koninklijke Bibliotheek, Den Haag, 1990.

[62] M. Sugie, O. Menzilcioglu, and H. T. Kung. CARGuide — on-board computer for automobile route guidance. Technical Report CMU-CS-84-144, Carnegie Mellon University, 1984.

[63] Rahul Sukthankar. RACCOON: A Real-time Autonomous Car Chaser Operating Optimally at Night. Technical Report CMU-RI-TR-92-13, Carnegie Mellon University, October 1992.

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

[65] Rahul Sukthankar. A description of SAPIENT's simulation environment. Unpublished document, September 1994.

[66] Rahul Sukthankar, Dean A. Pomerleau, and Charles E. Thorpe. Panacea: An active sensor controller for the ALVINN autonomous driving system. Technical Report CMU-RI-TR-93-09, Carnegie Mellon University, April 1993.

[67] Rahul Sukthankar, Dean A. Pomerleau, and Charles E. Thorpe. Panacea: An active sensor controller for the ALVINN autonomous driving system. In *Proceedings of International Symposium on Robotics Research*, 1993.

[68] Katia Sycara and Kazuo Miyashita. Learning control knowledge through case-based acquisition of user optimization preferences in ill-structured domain. In G. Tecuci and Y. Kodratoff, editors, *Machine Learning and Knowledge Acquisition: Integrated Approaches*. Morgan Kaufman, 1994.

[69] Charles E. Thorpe, 1994. Personal communication.

[70] Charles E. Thorpe and Jay Gowdy. Annotated maps for autonomous land vehicles. In *Proceedings of the DARPA Image Understanding Workshop*, 1990.

[71] 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.

[72] S. Tsugawa, T. Yatabe, T. Hirose, and S. Matsumoto. An automobile with artificial intelligence. In *Proceedings of 6th IJCAI*, 1979.

[73] 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.

[74] H. H. van der Molen and A. M. T. Botticher. Risk models for traffic participants: A concerted effort for theoretical operationalizations. In J. A. Rothengatter and R. A. de Bruin, editors, *Road Users and Traffic Safety*. Van Gorcum, Assen, 1987.

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

[76] Allen Waxman, Jacqueline J. LeMoigne, Larry S. Davis, Babu Srinivasan, Todd R. Kushner, Liang Eli, and Tharakesh Siddalingaish. A visual navigation system for autonomous land vehicles. *IEEE Journal of Robotics and Automation*, 3(2), April 1987.

[77] E. Weiss. *Untersuchung und Rekonstruktion von Ausweich und Fahrspurwechselvorgangen*. VDI-Verlag, 1987.

[78] Don Wharton. How to train your eyes for better driving. *Readers' Digest*, November 1958.

[79] Peter Wherrett. *Motoring Skills and Tactics*. Ure Smith, Sydney, 1977.

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

[81] Lotfi A. Zadeh. Neural networks and fuzzy systems: A dynamical systems approach to machine intelligence, 1991. Introduction in Bart Kosko's book.

[82] Jon K. Zell. Driver eye movements as a function of driving experience. Technical report, Department of Industrial Engineering, Ohio State University, June 1969.

[83] M. Zeller and R. Mock-Hecker. Modeling autonomous agents in a knowledge based simulation environment. In *Proceedings of the IEEE Conference on Tools with AI*, November 1993.