

A Fault Tolerant Control Architecture for Automated Highway Systems

John Lygeros, *Member, IEEE*, Datta N. Godbole, *Member, IEEE*, and Mireille Broucke

Abstract—A hierarchical controller for dealing with faults and adverse environmental conditions on an automated highway system (AHS) is proposed. The controller extends a previous control hierarchy designed to work under normal conditions of operation. The faults are classified according to the capabilities remaining on the vehicle or roadside after the fault has occurred. Information about these capabilities is used by supervisors in each of the layers of the hierarchy to select appropriate fault handling strategies. We outline the strategies needed by the supervisors and give examples of their detailed operation. In a companion paper details of communication protocols implementing some of these strategies are presented.

Index Terms—Automated highways, fault tolerance, hierarchical systems, large-scale systems, safety.

I. INTRODUCTION

INTELLIGENT vehicle highway systems (IVHS's) have been an active research area within the intelligent transportation systems (ITS) community for the past several years. One of the main objectives of the research in this area has been the development of an automated highway system (AHS) that will significantly increase safety and highway capacity without building new roads, by adding automation to the vehicle and the roadside. Several approaches to this problem have been proposed, ranging from autonomous intelligent cruise control (AICC) [1] (where the driver is in control of vehicle steering) to full automation supporting platooning [2]. An underlying assumption in most of the designs reported in the literature has been that the AHS operates under *normal conditions*. Roughly speaking, normal means benign environmental conditions and faultless operation of all the hardware, both on the vehicles and on the roadside. The only attempts to deal with degraded conditions ([3]–[6]) have mostly concentrated on specific faults rather than a general fault tolerant design framework.

Our goal in this paper is to propose an AHS design that will perform well under most conditions.¹ A common practice when

Manuscript received July 29, 1996. Recommended by Associate editor, J. Winkelman. This work was supported by the California PATH program, Institute of Transportation Studies, University of California, Berkeley, under MOU-135, 288, 312, and 319.

J. Lygeros and M. Broucke are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720-1770 USA (e-mail: {lygeros; mire}@eecs.berkeley.edu).

D. N. Godbole is with the Honeywell Technology Center, Minneapolis, MN 55418 USA (e-mail: godbole_datta@htc.honeywell.com).

Publisher Item Identifier S 1063-6536(00)01780-2.

¹The only degraded conditions that we do not consider are faults in the design or the implementation of the software (for example, deadlocks in the communication protocols). We assume the design will be verified before being implemented.

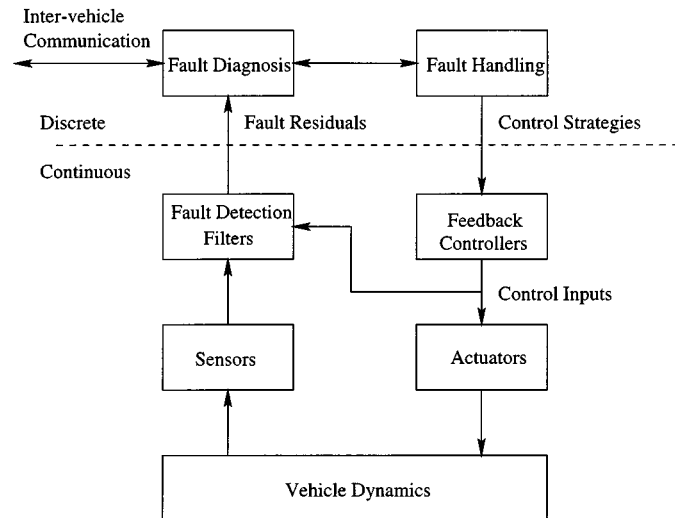


Fig. 1. Fault management system.

designing such *fault tolerant* control schemes is to make use of two modules (Fig. 1): a *fault detection module*, to determine whether a certain fault has occurred and a *fault handling module*, where special controllers are implemented to minimize the impact of the fault on the system performance. Because the system performance is likely to degrade anyway, we will use the term *degraded modes of operation* to describe operation under these special controllers. The extended control scheme should guarantee graceful and gradual degradation in performance.

Detection of failures in an AHS is a very challenging problem. Fault detection filters can be designed [7]–[9] to identify faults in the on-board sensors and actuators. Due to the distributed, multiagent character of the AHS problem, communication with neighboring vehicles may also be required (in addition to the fault detection filters) for complete diagnosis and isolation of faults [10], [11]. In this paper we assume that the fault detection module has already been designed and propose a design for the fault handling module. It will become apparent that, even in this restricted setting, the task is still formidable. We give an overview of what is involved, establish a framework for partitioning the task into more manageable subproblems, and formalize the requirements that the solution to each of them will need to satisfy. In related work (see for example [12]) we provide the details for the solution.

The contribution of this paper is twofold. First, a systematic method for extending normal mode control hierarchies to make them fault tolerant is presented. The method is fairly general, and is applicable to other large scale, multiagent systems, such as air traffic management or uninhabited aerial vehicles.

Second, we present innovative designs of new maneuvers and control strategies for dealing specifically with fault handling on an AHS.

This paper is arranged in five sections. In Section II we give an outline of the design process. We discuss the procedure we followed to produce the fault tolerant design and highlight the implications that fault tolerance imposes on the hierarchical structure. To fix the terminology we also describe briefly the normal mode control hierarchy of [2], that forms the starting point for our design. In Section III we present the first step of the design process, a monitoring scheme to track the capability of the system in the presence of faults and discuss the fault classification that this monitoring scheme induces. In Section IV we briefly discuss the new controllers that need to be designed to deal with these fault classes. Finally, in Section V we highlight some of the issues raised by the design process.

II. OUTLINE OF PROPOSED SOLUTION

A. Overview of Normal Mode Control Hierarchy

Our framework builds on the control hierarchy proposed in [2] for normal operation of a fully automated highway system that supports *platooning* of vehicles. The platooning concept assumes that traffic on the highway is organized in groups of tightly spaced vehicles (platoons). The first vehicle of a platoon is called the *leader*, while the remaining vehicles are called *followers*; a platoon consisting of a single vehicle is called a *free agent*. Spacing among the followers is assumed to be tight (of the order of 1–5 m). Platoon leaders on the other hand are assumed to maintain a large spacing from the platoon ahead (of the order of 30–60 m). Recent theoretical, numerical, and experimental studies have shown that an AHS that supports platooning is not only technologically feasible but, if designed properly, may lead to an improvement of both the safety and the throughput of the highway system, under normal operation [13]–[15].

Implementation of the platooning concept requires automatic vehicle control, as human drivers are not fast and reliable enough to produce the necessary inputs. To manage the complexity of the design process a hierarchical controller is proposed in [2]. The controller is organized in four layers (Fig. 2). The top two layers, called *network* and *link*, reside on the roadside and are primarily concerned with throughput maximization, while the bottom two, called *coordination* and *regulation*, reside on the vehicles² and are primarily concerned with safety. The network layer is responsible for the flow of traffic on the entire highway system, for example, several highways around an urban area. Its task is to prevent congestion and maximize throughput by dynamically routing traffic. The link layer coordinates the operation of sections (links) of the highway (for example the highway segment between two exits). Its primary concern is to maximize the throughput of the link. With these criteria in mind, it calculates an optimum platoon size and an optimum velocity and decides which lanes the vehicles should follow. It also monitors incidents and diverts traffic away from them, in an attempt to minimize their impact

²The *physical layer* is not part of the controller. It contains the plant, i.e., the vehicles and highway, with their sensors, actuators, and communication equipment.

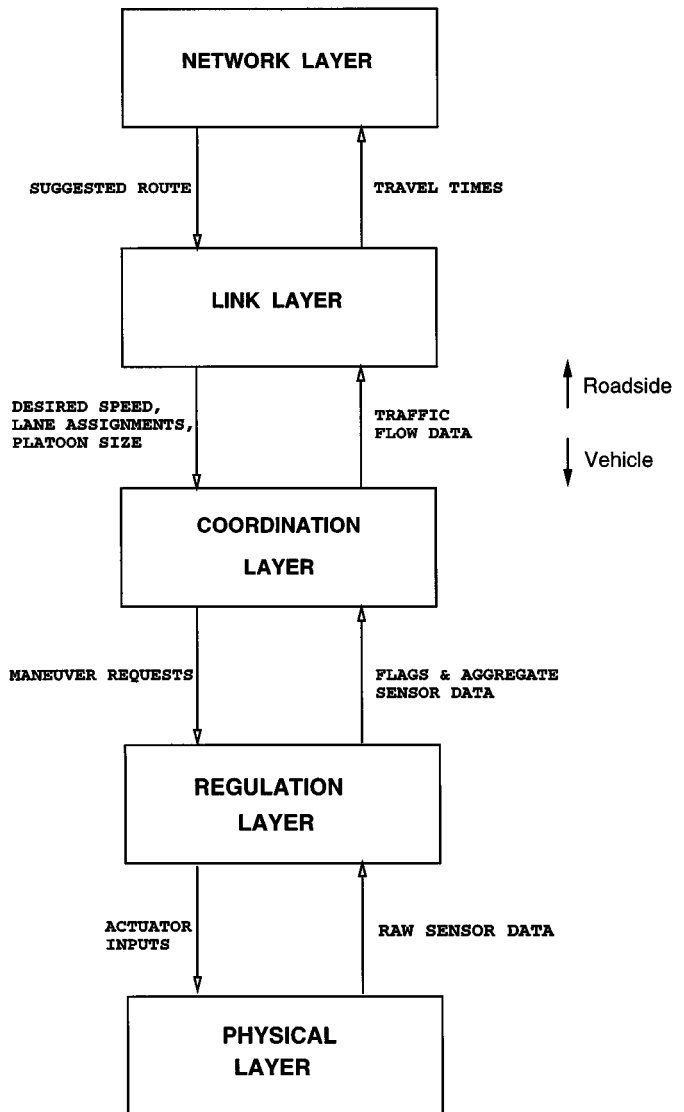


Fig. 2. Normal mode AHS control hierarchy.

on traffic flow. A number of designs have been proposed for the link layer, that make use of traffic flow models [3], [16] the concept of highway work [17], or the concept of highway space-time [18] to model the traffic.

The coordination layer coordinates the operation of neighboring platoons. It receives the link layer commands and chooses specific maneuvers that the platoons need to carry out. For normal operation, these maneuvers are *join* to join two platoons into one, *split* to break up one platoon into two, *lane change*, *entry* and *exit*. The design of [19] and [20] uses communication protocols, in the form of finite state machines, to organize these maneuvers in a systematic way. The regulation layer receives the coordination layer commands and translates them to throttle, steering, and braking input for the vehicle actuators. For this purpose it utilizes a number of continuous time feedback control laws (see, for example, [20]–[25]) that use the readings provided by the sensors to calculate the actuator inputs required for a particular maneuver. In addition to the control laws needed for the maneuvers, the regulation

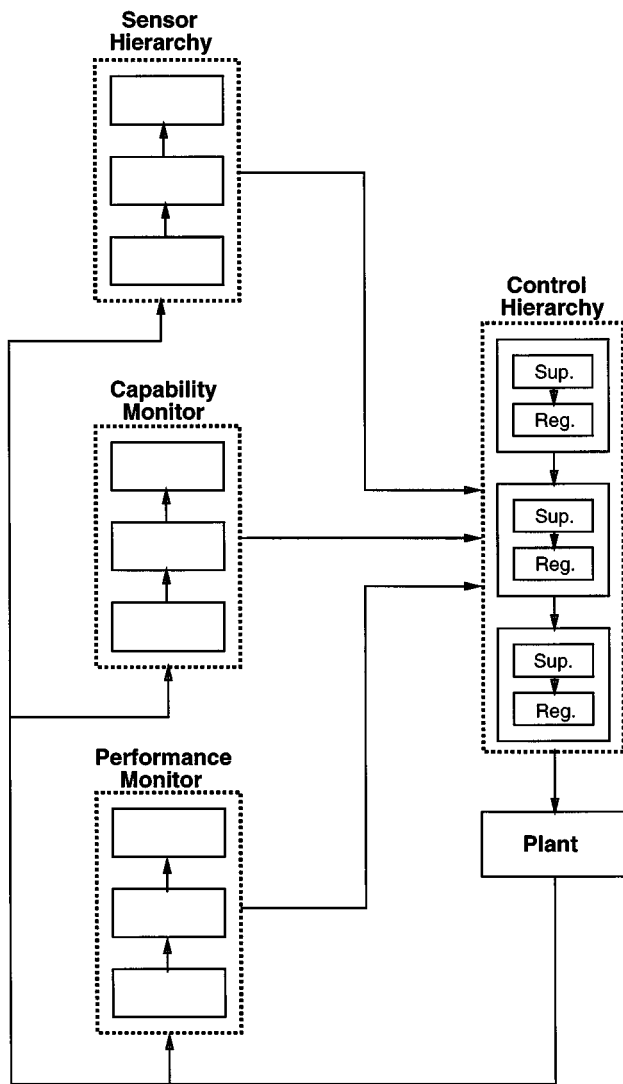


Fig. 3. Fault tolerant controller hierarchy.

layer makes use of two “default” controllers, one for *leader* and one for *follower* operation.

Among all the proposed AHS concepts [26], platooning is the one with the highest degree of automation and centralization. For this reason, it is also the one most prone to catastrophic failures. Although the normal mode hierarchy and the control laws discussed here are mainly designed for a platoon-based AHS, some of them can easily be extended to other AHS concepts, and even to partially automated systems. The main obstacles to extending our design to other concepts are that we ignore the possible interaction between the human operator and the vehicle and that some of the fault tolerant maneuvers assume the presence of intervehicle communication.

B. Extended Hierarchy

The overall structure of the proposed fault tolerant hierarchical controller is shown in Fig. 3. It is clearly more complicated than the normal mode control hierarchy. The reason is that the controller for normal operation can assume that the capability of the system is fixed. Under this assumption, the only information

that the controller needs is the state of the physical process, typically collected through sensors. Because the control hierarchy involves different modeling languages at each level, the state information needs to be processed and presented to the controller at the appropriate level of abstraction. It is convenient to think of this processing as being carried out by a *sensor hierarchy*, a separate hierarchical arrangement that operates alongside the control hierarchy.

The constant capability assumption is no longer valid in the presence of faults and extreme environmental conditions. Faults induce discrete qualitative changes in the system dynamics, that dictate discrete transitions in the control scheme. Information about these discrete changes has to be processed and propagated to the appropriate levels of the control hierarchy, that need to take action in response to the faults. It is convenient to think of this task as being carried out by a *capability monitor*, a hierarchical arrangement that collects the fault detection information and feeds it to the appropriate level of the hierarchy.

Extreme environmental conditions lead to gradual, quantitative changes in the system dynamics. The effect of such changes is continuous (as opposed to discrete) degradation in the system performance. This can eventually lead to discrete changes in the control scheme if, at some point, the degradation is severe enough so that the performance specifications can not be met. Therefore, the information processing in the case of extreme environmental conditions is more closely coupled to the controller structure. None the less it may still be convenient to think of a special hierarchical structure, the *performance monitor*, that collects the environmental information and determines how it affects each level of the control hierarchy.

Finally, under normal operation, the strategy of the controller is fixed. This strategy may involve switching among various controllers and control objectives; however, the switching patterns are predetermined and the switching depends only on the state of the physical process. Once the system capabilities start changing, however, the strategy may also have to be modified: control objectives may have to be dropped, certain controllers may become inoperable or inefficient, etc. Switching between strategies takes place at a higher level than switching between controllers for a fixed strategy. It is convenient therefore to think of this metaswitching as being controlled by distinct levels of the control hierarchy. Each one of the original hierarchy levels can be split into two layers. The top layer, which we call the *supervisor*, receives the system capability information and switches between strategies accordingly. The lower layer, the *regulator*, is responsible for implementing the chosen strategy.

C. Proposed Design Process

The fault tolerant control hierarchy was developed in a series of steps, that involved:

- 1) identification of faults and causes of gradual performance degradation;
- 2) development of a framework for modeling the capability of the system and the effect of the factors identified in Step 1) on it;
- 3) classification of the factors in Step 1) according to their effect on system capability;

- 4) extension of the control hierarchy to deal with the fault classes established in Step 3);
- 5) design of controllers for the extended hierarchy;
- 6) verification (wherever possible) and simulation of the extended hierarchy;
- 7) identification of the shortcomings of the proposed design, and redesign.

Many iterations between Steps 4)–7) are needed before a satisfactory design is obtained. Steps 2) and 3) are the main topic of this paper. An exhaustive list of faults and other causes of performance degradation [Step 1]), compiled by interviewing researchers in the California PATH project, can be found in [27]. A framework for modeling the capability of the system in the presence of faults [Step 2)] is presented in Section III. The fault classification [Step 3)] induced by our framework is discussed in Section III-C. Based on the work carried out for the first three steps, in Section IV we discuss the requirements that the extended controllers need to satisfy [Step 4)]. More details on the design, as well as verification results [Steps 4)–6)] can be found in the companion paper [12] and in [5] and [28]–[30].

III. MODELING THE SYSTEM CAPABILITY

A. Capability Monitor

The control scheme for normal operating conditions presented in [2] relies on a number of sensors, actuators, and communication devices, both on the vehicles and on the roadside. All this additional hardware, as well as the standard mechanical and electronic components of the vehicles, are prone to failure. In this paper the faults are modeled as discrete events and the capability monitor used to determine their impact on the system is modeled by a hierarchy of predicates. Each predicate monitors a single functional capability and returns a one (true) if the system possesses the capability in question and a zero (false) otherwise.³ The predicates are arranged in a hierarchy similar to that of the normal mode control hierarchy. The values returned by the higher layer predicates depend on the values of the lower layer predicates. This scheme is then used to systematically go through combinations of faults and design specialized control laws that utilize the remaining capabilities, so that the impact of the faults on the system is minimized.

1) *Physical Layer Predicates:* We assign a predicate to each one of the sensor, actuator, and communication resources. Initially, all the physical layer predicates return one. A fault is modeled as a discrete event that turns the value of the corresponding predicate to zero. Assuming that the control scheme requires n_a actuators, n_s sensors, and n_c communication devices, the capability of the physical layer can be thought of as a vector c_P of zeros and ones of dimension $n_s + n_a + n_c$

$$c_P \in C_P = \{0, 1\}^{n_s+n_a+n_c}.$$

³If the fault detection module returns probabilities of failures instead of discrete fault events, the capability structure presented here can be modified to propagate the probabilistic information through the hierarchy. A probabilistic capability monitor allows one to set thresholds on proper functioning of a particular maneuver which may be easier than setting thresholds on the fault detection module for individual fault signatures.

For simplicity, the actuator predicates are assumed to reflect the capability of the vehicle to accelerate, decelerate, and turn. Therefore they incorporate information about basic vehicle functionality, like engine and tires being in proper working order. Predicates for these basic functionalities can explicitly be added at the cost of a small increase in the complexity of the monitor. Similarly a sensor predicate reflects the ability of a vehicle to sense its environment. A more complicated structure for the sensor predicates can be constructed to reflect things like physical sensor redundancy.

2) *Regulation Layer Predicates:* The regulation layer contains controllers for controlling the *longitudinal* (along the lane) and *lateral* (across the lane) vehicle motion. Each one of these controllers makes use of physical layer resources, primarily sensors and actuators. For a regulation layer controller to be functional, all of these resources need to be available. Therefore, the availability of a regulation layer controller can be modeled by a predicate whose value depends on the values of the predicates for the physical layer.

Consider, for example, the longitudinal controller proposed in [24] for the leader of a platoon. This controller uses sensor readings for the velocity and acceleration of the vehicle and for the spacing and relative velocity with respect to the preceding vehicle to calculate inputs for the throttle and brake actuators. Without getting into the details of the control law, we can see that the lead controller predicate can be viewed as an *AND* of the predicates for the velocity, acceleration, spacing, and relative velocity sensors and the brake and throttle actuators. The controller proposed for the followers in a platoon in [31], on the other hand, makes use of additional information about the state of the leader of the platoon. It is envisioned that this information can be transmitted to all the followers using an infrared communication link. Therefore, the predicate for the longitudinal follower law should also depend on the predicate for the infrared communication link.

If there are n_{long} longitudinal laws and n_{lat} lateral laws, then the capability of the regulation layer can be thought of as a vector of zeros and ones of dimension $n_{long} + n_{lat}$. The design of the control laws implies a mapping from the vector coding the capabilities of the physical layer to the vector coding the capabilities of the regulation layer

$$F_R: C_P \longrightarrow C_R = \{0, 1\}^{n_{long}+n_{lat}}.$$

Fig. 4 shows this mapping for the controllers of [20], [22], [24], [25], and [31].

3) *Coordination Layer Predicates:* From the point of view of the coordination layer, the regulation layer control laws represent resources that can be used to carry out maneuvers. Each maneuver will need to make use of two control laws, one longitudinal and one lateral. For the coordination layer to be able to invoke a maneuver, both control laws should be operational. For example, for the coordination layer to command a platoon leader to join, at least one (of possibly many) longitudinal join law and one lateral lane keeping law should be operational.

Let n_{man} denote the number of maneuvers that may be requested by the coordination layer. Then the system capability at the regulation/coordination layer interface can be modeled by a

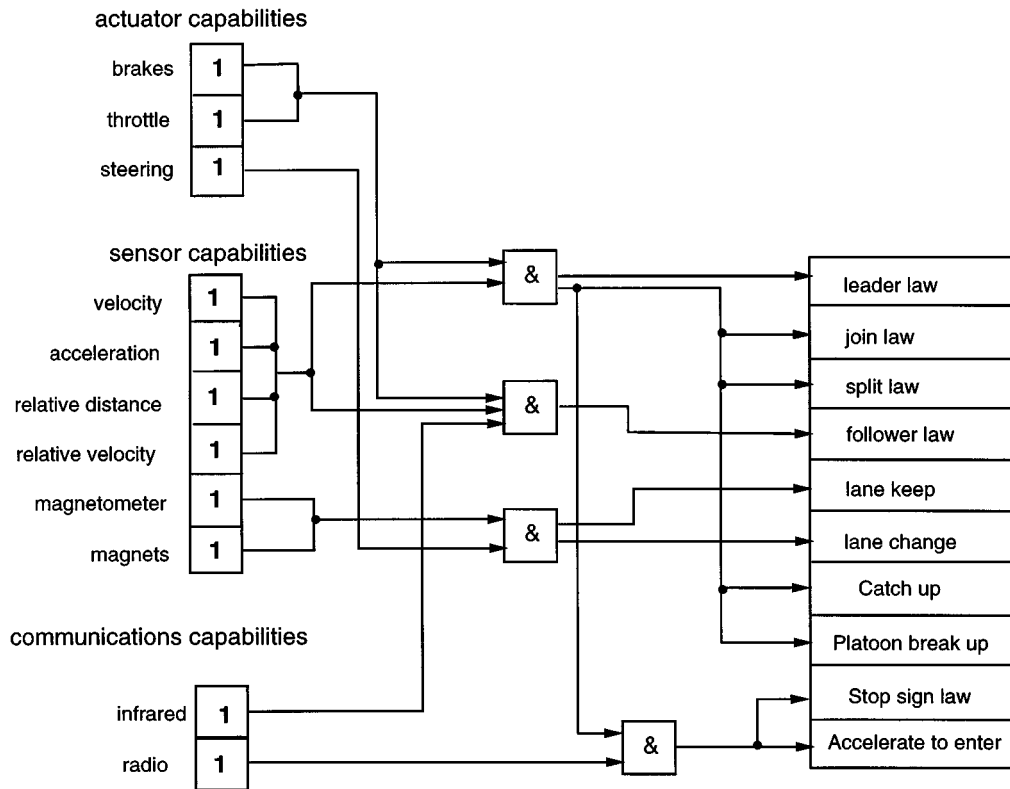


Fig. 4. Physical and regulation layer capabilities.

vector, c_I , of zeros and ones of dimension n_{man} . The design of the interface induces a mapping

$$F_I: C_R \longrightarrow C_I = \{0, 1\}^{n_{man}}.$$

For the maneuvers of [19] and [20], the controllers of [20], [22], [24], [25], [31], [32], and the interface of [33], the map F_I is shown in Fig. 5. To execute the protocols that organize the maneuvers, the coordination layer needs to be able to communicate with neighboring vehicles. Therefore, the capability of the coordination layer to operate in its normal mode can be expressed as a predicate on the values of the capability vector for the interface and the communication device predicates (Fig. 5).

From the figures it should be clear that if a fault damages any of the vehicle's basic functions (toggling a physical layer predicate to zero) the normal mode of the coordination layer is likely to be rendered inoperable. For this purpose additional coordination layer strategies that are capable of operating in these reduced circumstances need to be designed. These new strategies may require additional maneuvers and regulation layer control laws that try to make the best of the remaining capabilities of the vehicle. As we shall see in Section IV-B, some of these maneuvers will require close cooperation with the neighboring vehicles. Therefore, the applicability of the degraded mode coordination strategies can be expressed as a predicate on the values of the interface capability vector and the communication device predicates of the faulty vehicle, as well as the interface capability vectors of the neighboring vehicles.⁴ Once all degraded

⁴It is assumed that knowledge about the capabilities of the neighbors will be obtained once communication has been established.

mode strategies have been designed the coordination layer capability can be expressed as a vector of zeros and ones. The dimension of this vector will be equal to the number of strategies. The design of the strategies induces a mapping from the values of the capability predicates for the interface of the faulty vehicle, the interface its neighbors, and the communication devices, to the capability predicates for the coordination layer strategies. Maps similar to the one shown in Fig. 5 can be constructed for the emergency strategies introduced in Section IV-B.

It should be noted that the coordination layer predicate structure illustrates the explicit separation between supervisor and regulator discussed in Section II. The interface predicates can be thought of as the capability of the regulator part of the coordination layer, while the strategy predicates can be thought of as the capability of the supervisor part.

4) *Link Layer Predicates*: The link layer controller makes use of information about the density and average velocity of traffic to produce commands for the vehicles that locally maximize throughput and equalize lane usage. The traffic information is obtained by roadside sensors and the link layer control commands are broadcast to the vehicles via radio. To improve the resolution of the information and the commands, most proposed link layer designs [3], [18] partition the highway into sections a few meters long (of the order of 500) and one lane wide. Entrance and exit lanes are also treated as separate sections. Link layer controllers of adjacent sections exchange traffic and control information by a wire line communication network.

A link layer controller for degraded conditions of operation needs information about situations that limit the capability of traffic in its sections. The example presented in Section IV-A

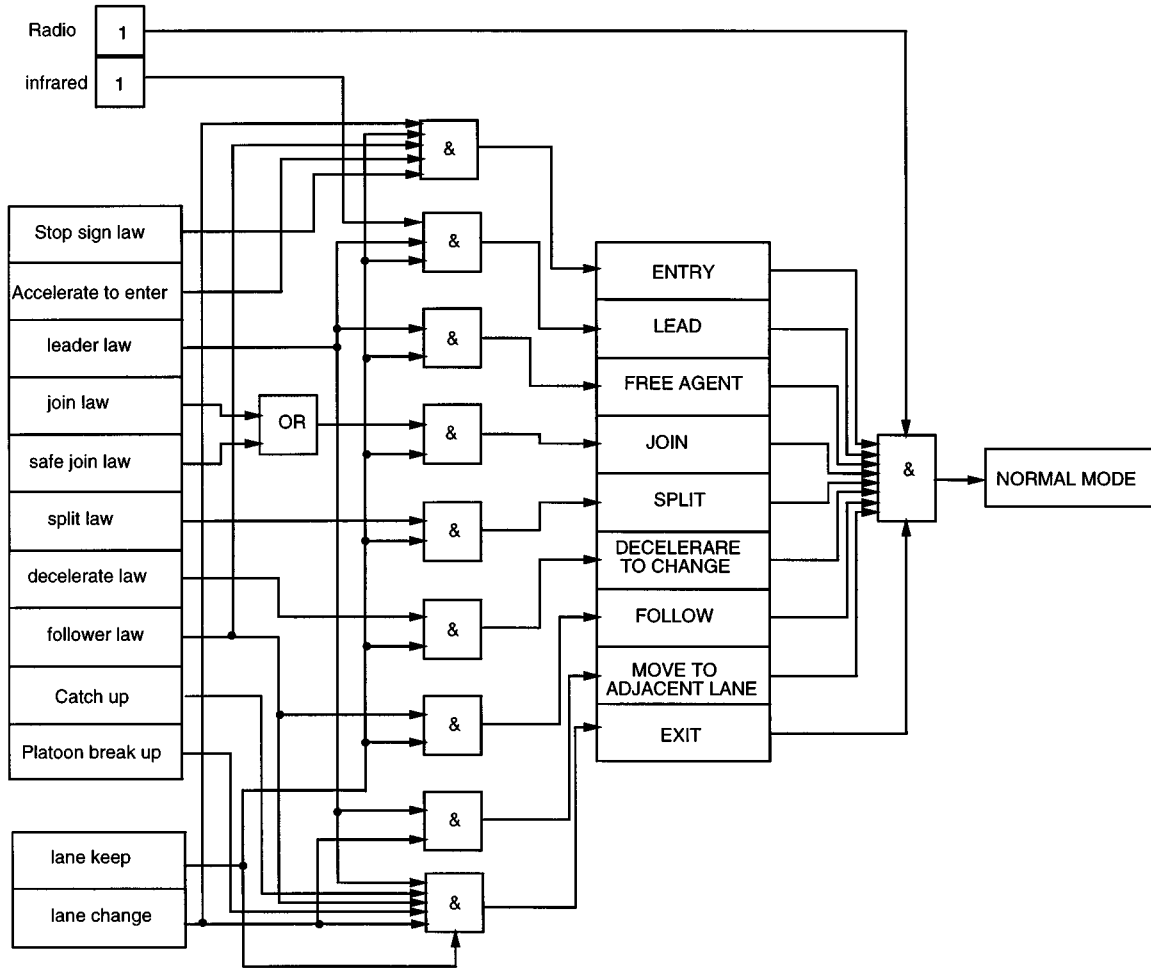


Fig. 5. Regulation and coordination layer capabilities.

indicates four such situations: *section blocked*, *section contains no vehicles*, *section contains vehicles queued behind an incident* and *section contains emergency vehicles*. These properties can be modeled as a set of predicates for each section, that return one if the link possesses the property (e.g., is blocked) and zero otherwise. The value returned by these predicates depends on the capability vectors of all vehicles in the section. For example, if a section contains a stalled vehicle then the predicate *section blocked* returns one. In addition the values of the predicates should also reflect infrastructure faults such as loss of roadside sensing capability, or loss of vehicle to roadside communication capability.

B. Performance Monitor

The performance monitor involves three elements. The first is the *causes of gradual performance degradation* which the controller will have to guard against. They include adverse weather conditions (such as rain, fog or snow) and gradual hardware degradation (such as brake wear). We will use \mathcal{C} to denote the set of performance degradation causes. Each element of \mathcal{C} is assumed to be a real number whose magnitude signifies the

severity of the cause (e.g., the longitudinal wind measured in meters per second).⁵

The second element is the *performance parameters* that can be used to monitor the system performance. These performance parameters depend on the layer of the hierarchy and include, for example, the maximum and minimum deceleration available to the vehicle (for the physical layer) and the maximum tracking error of the continuous controllers (for the regulation layer). We use \mathcal{P} to denote the set of performance parameters. The set \mathcal{P} is divided according to the level of the hierarchy associated with each parameter

$$\mathcal{P} = P_P \cup P_R \cup P_C \cup P_L \cup P_N$$

where P_P are the parameters associated with the physical layer and P_R the parameters associated with the regulation layer, etc. The elements of \mathcal{P} are again assumed to be real valued.

The final element is the *performance requirements*. They can be thought of as thresholds on the performance parameters.

⁵In its simplest form an element of \mathcal{C} can be thought of as a predicate that returns one if a cause is present and zero if it is not. Soft computing approaches, such as fuzzy logic, may be used to quantify more elusive causes, such as snow or fog.

More formally performance requirements are predicates on the space of performance parameters

$$R_i: \mathcal{P} \longrightarrow \{\text{True}, \text{False}\} \quad i = 1, \dots, r$$

where r denotes the number of performance requirements that we consider.

The design of the performance monitor involves finding relationships between causes of gradual performance degradation and the performance parameters. It should be noted that the performance parameters of higher layers will depend on those of lower layers (e.g., the tracking errors depend on the acceleration bounds). However, there can be no loops in these dependencies, i.e., lower layer parameters do not depend on higher layer ones. Therefore, the overall relationship can be flattened into a map

$$f: \mathcal{C} \longrightarrow \mathcal{P}$$

that determines how the causes of performance degradation affect the performance parameters. This map will depend on the details of the control laws. Qualitatively the dependencies will be fixed (unless major changes are made in the hierarchy) even though the map will change quantitatively with any change in the controller parameters. In this framework, the range of conditions $\hat{\mathcal{C}}$ under which the performance of the system is acceptable is given by

$$\hat{\mathcal{C}} = \bigcap_{i=1}^r f^{-1}(R_i^{-1}(\text{True})) \subset \mathcal{C}.$$

Many iterations may be needed to properly capture the system requirements in terms of the above equation for $\hat{\mathcal{C}}$. Enhancing the robustness of the system involves enlarging $\hat{\mathcal{C}}$. Our framework can be used for off-line robustness enhancement, where the controllers are tuned to accommodate a larger set of conditions $\hat{\mathcal{C}}$. The framework can also be used to increase the system autonomy by on-line tuning of the controllers.

Even after the domain $\hat{\mathcal{C}}$ has been maximized, there will still be conditions in $\mathcal{C} \setminus \hat{\mathcal{C}}$ which are not covered. These conditions for which performance is unacceptably degraded will be treated in a way similar to the treatment of loss of capability due to faults. In this sense, the effects of gradual degradation and limits of robustness can be modeled as an extra term on the predicates of the capability monitor. A more detailed discussion of this process can be found in [27], where a list of performance parameters is introduced, the effect of the causes on the parameters is specified (in most cases qualitatively), and detailed examples of the robustness enhancement and degraded mode initiation process are given. Here we illustrate the process by an example.

Consider the longitudinal leader control law, a regulation layer performance parameter, the tracking error between the actual and the desired spacing ($P_R = \{e\}$) and a physical layer performance parameter, the minimum acceleration that can be applied by the vehicle ($P_P = \{a_{\min}\}$). P_R depends on P_P , which in turn depends on a number of causes of performance degradation, such as the condition of the road and the tires. Consider two performance requirements

$$R_1 = \{a_{\min} < -3\} \quad R'_1 = \{a_{\min} \in [-3, -1]\}.$$

R_1 is used for the normal mode, which requires the deceleration to be at least -3 m/s^2 . If some C_i (for example rain) causes a_{\min} to become greater than -3 , but less than -1 , the control laws are augmented (for example the nominal spacing is increased), inducing a change in f . The requirement is also modified to R'_1 , to reflect the change. If some other C_i (for example leak in brake fluid) causes a_{\min} to become greater than -1 , the robustness module calls for a degraded mode, by toggling the predicate for the brake actuator to zero.

C. Fault Classification

Because the number of faults that need to be considered is large, we would like to be able to design controllers that deal with entire classes of faults or combinations of faults. The capability and performance structures can be used to induce such a classification. Faults in the same class lead to the same predicates in the capability structure returning zeros and therefore can be handled by similar controllers. An additional advantage is that if some faults have been overlooked by the current design they can be easily introduced later on, by determining the class in which they belong. We distinguish the following classes.

- 1) *Vehicle Stopped/Must Stop*: This class contains the most severe faults, such as faulty steering or engine failure. The vehicle can not continue moving on the AHS safely and has either already come to a stop or should be commanded to do so and wait to be towed away. Because of the severity of the situation, all the layers of the control hierarchy up to the link layer undergo some degradation in performance and assist in resolving the fault condition. Therefore predicates all the way up to the link layer are affected. The class can be divided into subclasses, depending on the action necessary to bring the faulty vehicle to a stop (e.g., severe versus mild braking). Once the vehicle has come to a stop, algorithms are used in the link and coordination layer, to assist in removing the stalled vehicle and to divert neighboring traffic.
- 2) *Vehicle Needs Assistance to Exit*: The faults in this class are slightly less severe, for example a stuck transmission or inability to sense vehicles in adjacent lanes. The vehicle may continue moving but must exit the AHS as soon as possible.⁶ Moreover, it needs the assistance of its neighbors to do so. Faults in this class will result in the normal mode coordination layer predicate returning a zero without any of the link layer predicates being affected. Therefore, these faults can be handled locally, by specialized coordination and regulation layer strategies.
- 3) *Vehicle Needs no Assistance to Exit*: The faults in this class are even less severe, for example a single fault in a redundant array of sensors. Typically the vehicle is fully functional but should leave the system to avoid further problems (in case a second fault occurs for example). Faults in this class result in regulation layer predicates turning to zero, without any coordination layer predicates being affected. They are handled by special controllers in

⁶Here we do not assume the existence of a breakdown lane. If such a lane exists, then it may suffice to take the faulty vehicles there and wait for emergency assistance. The maneuvers and control laws developed here can be easily adapted to accommodate a breakdown lane.

the regulation layer and neither the neighboring vehicles nor the roadside need to be alerted.

- 4) *Vehicle Does Not Need to Exit*: This class contains minor faults (such as faulty headlights) that require no special action but should nonetheless be recorded and the driver should be notified in case he/she needs to alter their travel plans. Faults in this class result in physical layer predicates returning zeros, without any higher layer predicates being affected.
- 5) *Infrastructure Failures*: This class includes all faults that induce a reduction in the capability of the infrastructure, such as roadside sensor failures, roadside to vehicle communication failure, and roadside to roadside communication failure. Faults in this class result in link layer predicates returning zeros, without any changes in the coordination layer predicates. They typically lead to severe degradation in performance, but do not directly affect the safety of the vehicles. Some of them can be handled by the normal mode controllers of the link and network layers, but some may need drastic changes in the operation of the system.
- 6) *Driver—Vehicle Interaction Failure*: This class includes problems that occur during entry and exit to the AHS, when the driver has to either relinquish or resume control of the vehicle.⁷ These faults are resolved by simple additional strategies that do not interfere with the rest of the design (see [20] for details).

IV. CONTROLLER DESIGN

We now present specific suggestions for controllers to deal with each one of the faulty conditions. Recall that the controllers at each level of the control hierarchy consist of two layers, a supervisor that plans an appropriate strategy and a regulator that executes individual maneuvers to track this strategy (Fig. 6).

A. Link Layer

For normal operation, the primary consideration of the link layer is to maintain a smooth flow of traffic and ensure that all vehicles make their exits. Under degraded conditions, however, other considerations such as diverting traffic away from an incident and assisting emergency vehicles take precedence. To highlight this point consider the following example.

1) *Stalled Vehicle on a Multilane AHS*: Suppose a faulty vehicle has stopped in the middle lane of a three lane highway. The role of the link layer controller in this case will be to divert traffic away from the incident, assist vehicles queued up behind the incident to change lanes and facilitate the access of the emergency vehicles. The figures present a sequence of desired traffic patterns and the corresponding link layer commands for one possible strategy. Changes in the desired traffic pattern are triggered by changes in the link layer capability predicates.

In Fig. 7 the predicate *section blocked* of the section labeled *stop* changes to true. As a result, adjacent lanes are commanded to slow down to facilitate the vehicles from the blocked lane to change out. Some vehicles will queue up behind the incident.

⁷We assume that once on the freeway, the driver may not interfere with the system operation, except for route/destination selection.

The safety critical task of stopping vehicles before they collide with the stopped vehicle is carried out by their regulation layer control laws. In Fig. 8, the predicate *section contains no vehicles* becomes true for the section labeled L_2 in the blocked lane while the predicate *section contains queued vehicles* becomes true for the section labeled L_1 in the same lane. In response a gap is created in an adjacent lane by commanding upstream traffic to decelerate. The gap travels toward the stopped vehicles at the speed of the adjacent lane. As the gap approaches (Fig. 9), the queued up vehicles *Back Up* in the empty space in L_2 , speed up to the adjacent lane speed and change lane into the gap. The gap creation and vehicle removal will go on until either all the queued vehicles are cleared (predicate *section contains queued vehicles* for section L_1 becoming false) or emergency vehicles appear (predicate *section contains emergency vehicles* becomes true. Meanwhile, the emergency vehicles move toward the incident using the blocked lane (Fig. 10). As the lane is empty from L_2 onwards and the vehicles in this lane in L_3 are moving out, the emergency vehicle can move faster than vehicles in adjacent lanes. Alternatively, lane changes from the blocked lane to one of the adjacent lanes can be stopped to let that lane carry the emergency vehicle. Eventually, the emergency vehicle reaches the end of the queue and moves ahead of it (possibly using a gap created in the adjacent lane as above). In the recovery mode (Fig. 11) some restrictions on speed and lane changing activity are imposed to avoid collisions due to large velocity differentials across lanes. At this stage all the link layer predicates have returned to their normal values.

Note that the proposed strategy is such that the link layer does not play a safety critical role; its actions aim only to ease the congestion caused by the incident.

2) *Link Layer Supervisor*: The task of executing a fault handling strategy like the one discussed above is carried out by the link layer supervisor. The proposed strategy can easily be formalized by a finite-state machine, with states reflecting the stages of the strategy and transitions triggered by changes in the link layer capability predicates. Each stage corresponds to a different desired density and velocity profile for the traffic in the link. For example, Fig. 8 corresponds to a velocity profile with zero velocities in lane 2, sections L_1 and L_2 , small velocities in lanes 1 and 3, sections L_2 and L_3 and lane 2, section L_3 , and large velocities in lanes 1 and 3, section L_1 . The corresponding density profile has high densities everywhere, except lane 3, section L_2 and lane 2, section L_3 (where the density is low), and lane 2, section L_2 (where the density is zero).

The description of the strategy need not necessarily be in terms of density and velocity profiles. Such profiles are better suited if traffic flow models, such as the ones in [3], [16], and [30], are used. For other models [17], [18] description in terms of a desired distribution of vehicle activities may be more suitable.

3) *Link Layer Regulator*: The task of tracking the strategy determined by the supervisor is carried out by the link layer regulator. The objective at this level is to translate the desired traffic patterns to commands for the vehicles in each section. The commands will in general take the form of velocity, lane change, join and split recommendations; the precise format will depend on the modeling language used by the link regulator. The regulator

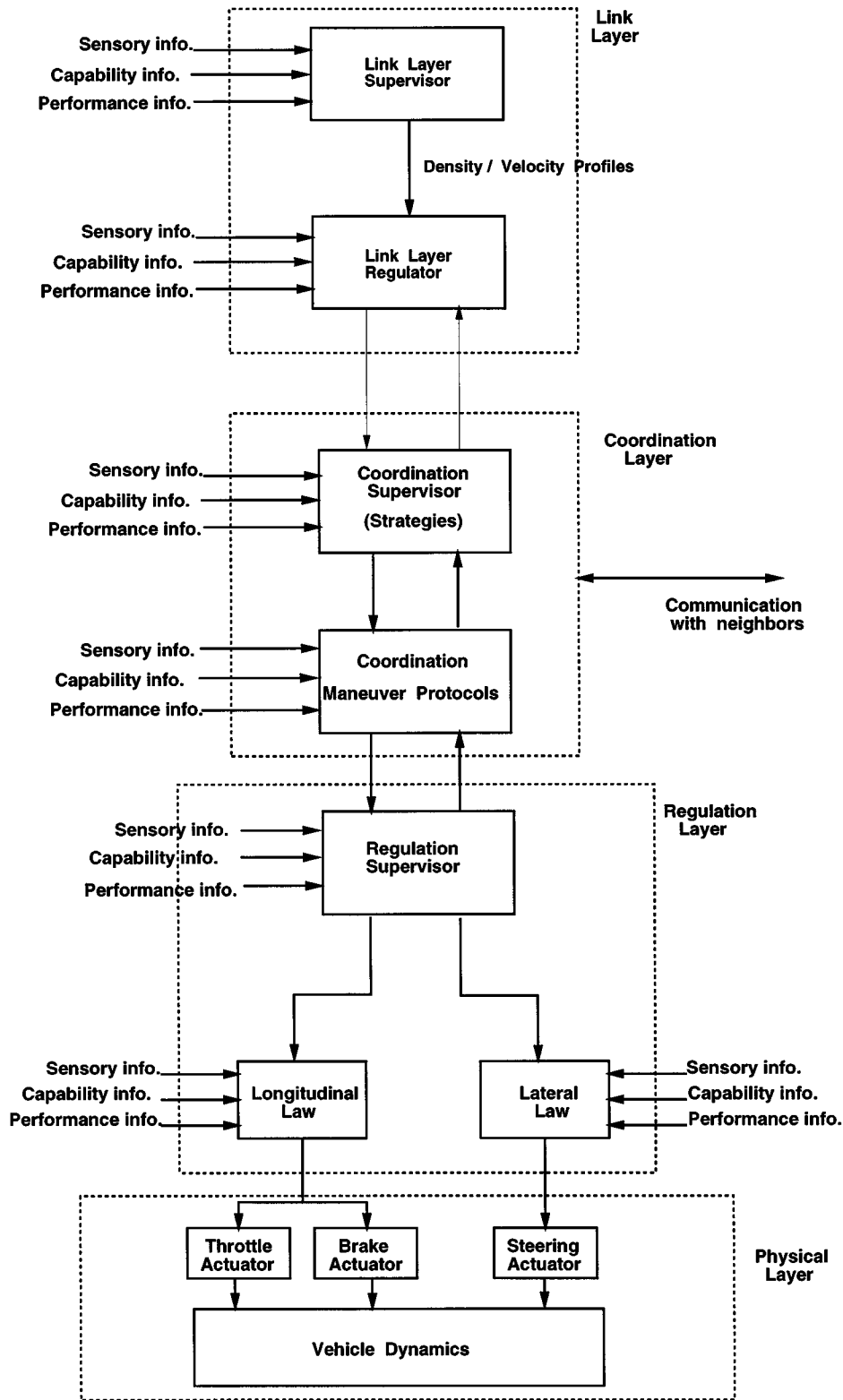


Fig. 6. Extended control architecture for degraded modes of operation.

should monitor the traffic in the link and use feedback to guarantee that the desired strategy is tracked. One possible regulator design is presented in [16]. The design accepts as inputs velocity and density profiles and issues velocity commands to the

vehicles (both longitudinal and lateral). Asymptotic tracking of the desired profiles is proved by Lyapunov analysis, assuming a flow model for the traffic. The performance of the regulator has been validated using the SmartPath [34] simulation platform.

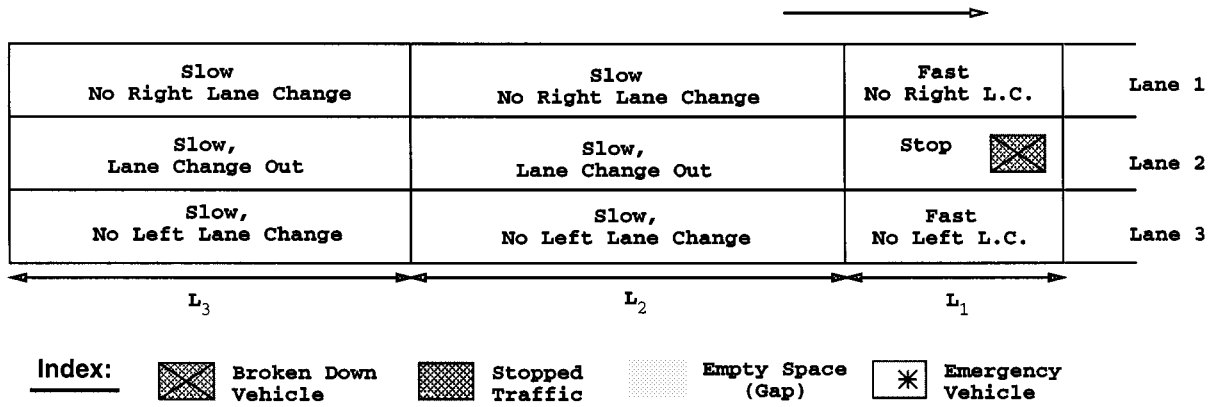


Fig. 7. Stage 1, vehicle stopped.

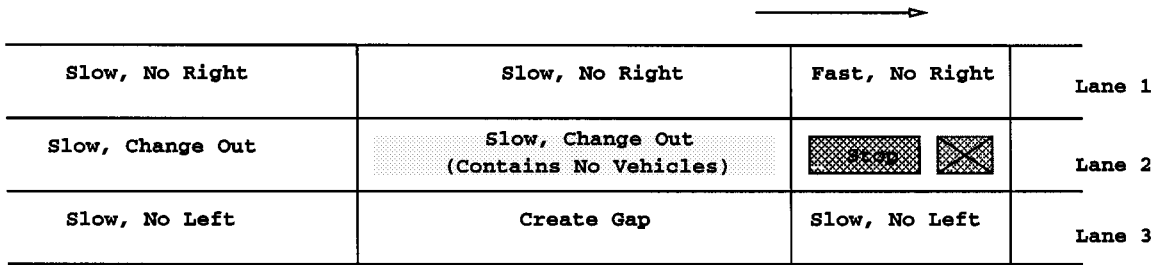


Fig. 8. Stage 2, queue buildup complete.

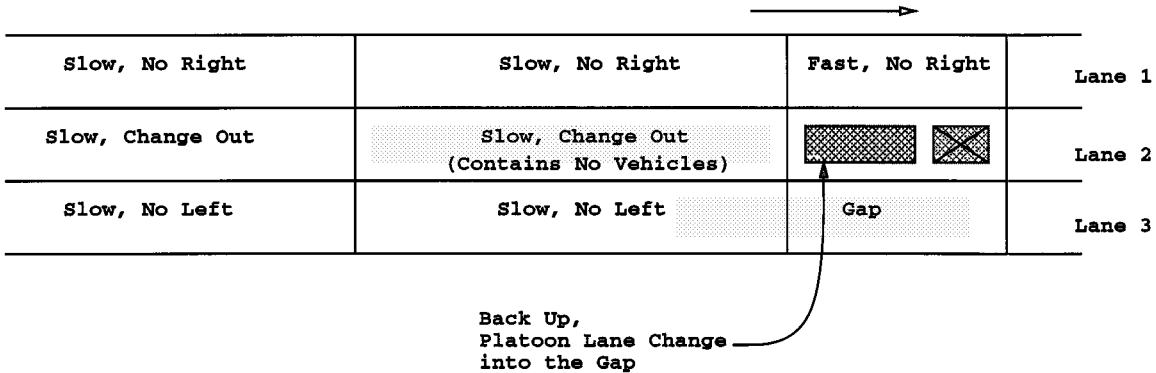


Fig. 9. Stage 3, queue dissipation.

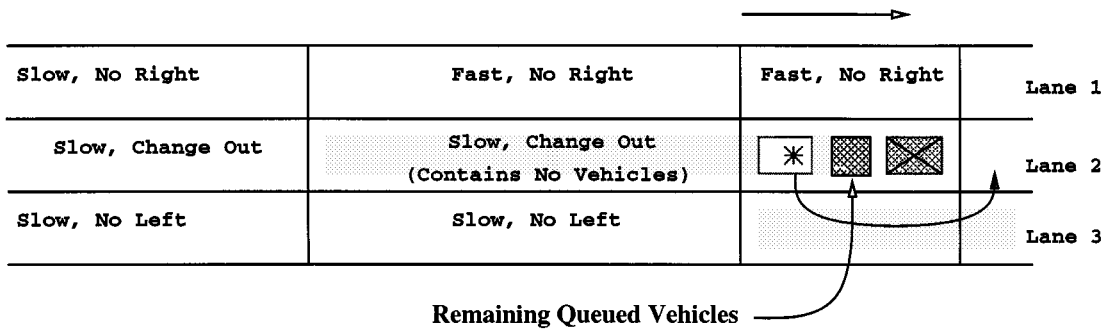


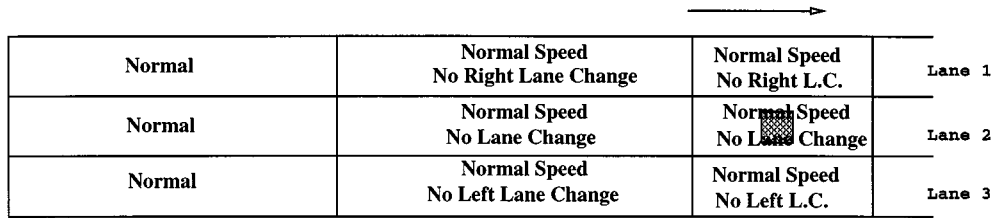
Fig. 10. Stage 4, emergency vehicle access.

B. Coordination Layer

The supervisor level of the coordination layer determines the sequence of maneuvers that a vehicle should carry out, while the regulator level contains protocols for coordinating individual maneuvers with neighboring vehicles. New strategies are added

both to the coordination supervisor and to the coordination regulator to extend the design to faulted conditions.⁸

⁸The extended coordination layer could also contain a map of the highway network to be used in case the vehicle can not obtain the broadcast information from the link layer, due to an infrastructure failure.



Normal	Normal Speed No Right Lane Change	Normal Speed No Right L.C.	Lane 1
Normal	Normal Speed No Lane Change	Normal Speed No Lane Change	Lane 2
Normal	Normal Speed No Left Lane Change	Normal Speed No Left L.C.	Lane 3

Fig. 11. Recovery.

1) *Coordination Layer Supervisor*: For faults in the class *vehicle stopped/must stop* three strategies are introduced to stop the vehicle: *Gentle Stop*, *Crash Stop*, and *Aided Stop*. For faults in the other classes, coordination layer strategies are designed to get the faulty vehicle out of the AHS as soon as possible. For this purpose three more strategies are introduced, *Take Immediate Exit* and *Take Immediate Exit—Escorted* for faults in the class *vehicle needs assistance to exit*, and *Take Immediate Exit—Normal*, for faults in the class *vehicle needs no assistance to exit*. During the execution of the degraded mode strategies, the faulty vehicle may request cooperation from neighboring vehicles. This cooperation can be encoded by means of communication protocols. A companion paper [12] describes the design and verification of these protocols in terms of interacting finite state machines. Here we only give a brief overview of the design.

- 1) *Gentle Stop and Crash Stop Strategies (GS, CS)*: are used by a faulty vehicle that is ordered to stop and can do so using its own brakes. *Gentle Stop* is used with faults that are not severe enough to require maximum deceleration (for example, engine or communication failures). The vehicle uses gentle braking to stop, to minimize the disturbance to the following vehicles. For *Crash Stop*, the severity of the fault dictates that the faulty vehicle should apply maximum deceleration (for example, steering faults and complete longitudinal sensor failures). Both the *Gentle Stop* and *Crash Stop* strategies do not require assistance from neighboring vehicles, therefore they are trivially implemented at the coordination regulator level (the strategies are also the maneuvers).
- 2) *Aided Stop Strategy (AS)*: is used by a vehicle with a “brakes off” failure. The faulty vehicle is aided by the vehicle immediately ahead of it in the same platoon to come to a stop. If the faulty vehicle is a leader, it uses the *Front Dock* maneuver to become a follower before executing the *Aided Stop* maneuver.
- 3) *Take Immediate Exit (TIE)*: (Fig. 12) is used for vehicles that must exit the AHS and can still operate as free agents (for example vehicles with infrared communication failures). The faulty vehicle (A) executes up to two *Forced Split* maneuvers (assisted by vehicles B and C) to become a free agent. It then executes a sequence of *Emergency Lane Change* maneuvers (assisted first by vehicle D and then by vehicle E) until it reaches the rightmost automated lane from where it takes the next exit.
- 4) *Take Immediate Exit—Escorted (TIE-E)*: (Fig. 13) is used by a faulty vehicle that has lost the capability to be a platoon leader but can still operate as a follower (for example,

after a longitudinal radar failure). In this case, the faulty vehicle A leaves the system as part of a two vehicle platoon with itself as the follower and an escorting vehicle B as a leader. This requires up to two *Forced Split* maneuvers (if the faulty vehicle is already initially a follower) or a *Front Dock* and possibly a *Forced Split* maneuver (if the faulty vehicle is initially a leader). Vehicle B escorts the faulty vehicle out of the AHS by executing sequence of *Emergency Lane Change* maneuvers of the two vehicle platoon. Once out of the AHS, the escorting vehicle drops off the faulty vehicle and reenters the AHS at the next entrance.

- 5) *Take Immediate Exit—Normal (TIE-N)*: is similar to the TIE strategy except the faulty vehicle uses the normal lane change and split protocols of [19] instead of *Emergency Lane Change* and *Forced Split*.
 - 6) *Normal (N)*: is the normal mode strategy of [2], implemented by means of finite-state machines in [19], and [20].
- 2) *Coordination Layer Regulator*: The strategies described above consist of sequences of maneuvers, which include the normal mode maneuvers of [19] and [20] as well as some new, emergency maneuvers. The emergency maneuvers needed to implement the above strategies are the following.
- a) *Forced Split (FS)*: is used by a faulty vehicle to become a free agent. If the faulty vehicle is a follower it requests the leader of the platoon to initiate a *Forced Split*. The leader divides the platoon at the desired location.
 - b) *Emergency Lane Change (ELC)*: is used by a free agent or a platoon. The faulty vehicle requests the leader of the platoon in the adjacent lane to create and maintain a gap so that the faulty vehicle can change lane into it.
 - c) *Front Dock (FD)*: (Fig. 14) is initiated by a platoon leader A that needs to join with the vehicle in front but can not safely execute a *Join* maneuver. The leader of the preceding platoon C orders the last vehicle in its platoon B to decelerate and close the gap between itself and the initiator. In the end, the initiator becomes the first follower of the new platoon.
 - d) *Aided Stop (AS)*: The aided stop maneuver forms the last part of the aided stop strategy. The faulty vehicle uses its engine to decelerate while the assisting vehicle (the vehicle immediately ahead of the faulty vehicle) applies gentle braking, lets the faulty vehicle collide with it and then uses its brakes to bring the combined mass of both vehicles to a stop.
 - e) *Queue Buildup and Queue Management (QB, QM)*: are used whenever a faulty vehicle is stopped. Vehicles in the same lane immediately behind the faulty vehicle will

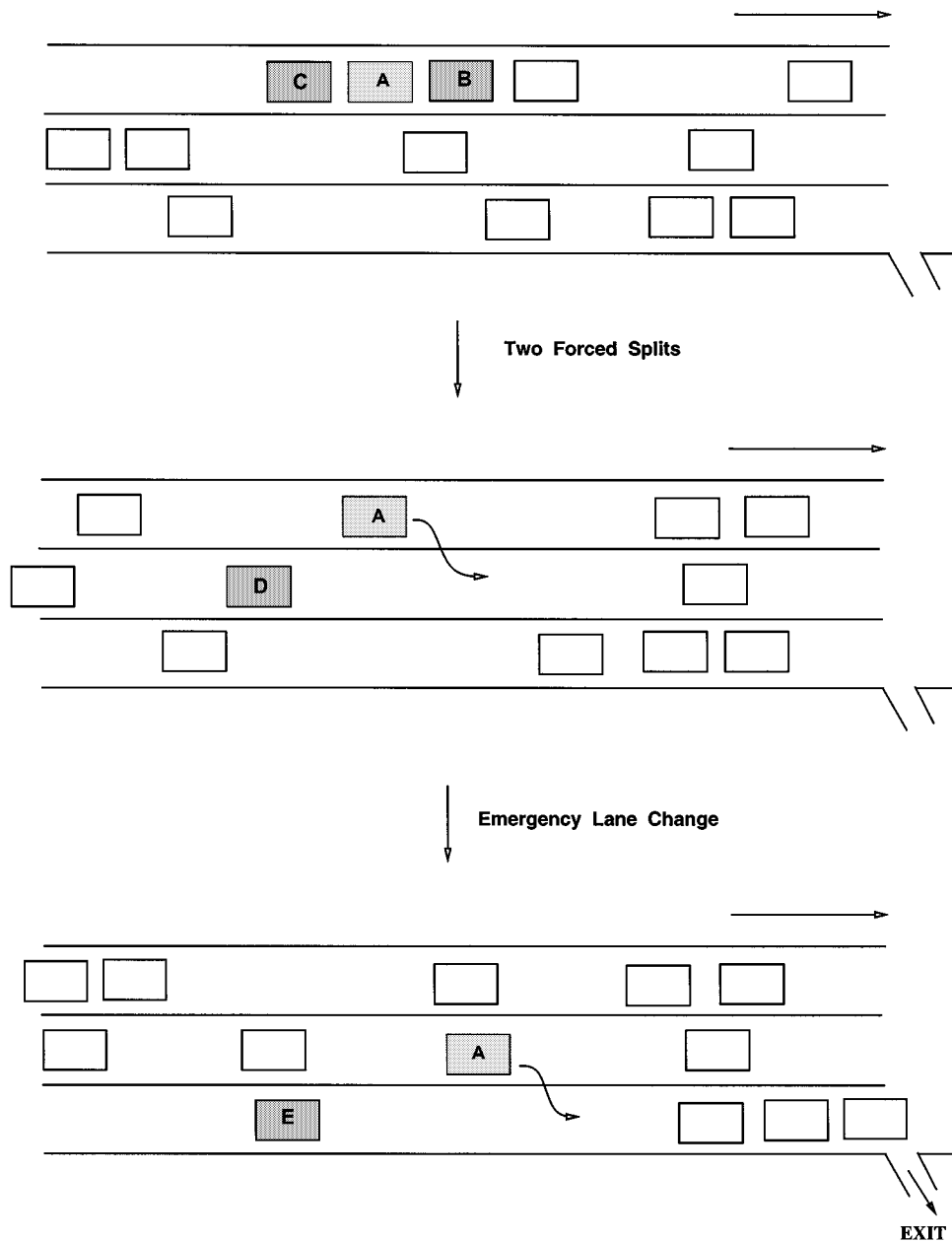


Fig. 12. Take immediate exit.

form a queue of stopped vehicles. The *queue buildup* maneuver is used to keep track of the number of vehicles in the queue and the identity of the last queued vehicle. For the queue buildup to stop, there must be a large gap behind the last vehicle of the queue. Once the queue buildup has stopped, the *queue management* maneuver is used to dissipate the queue in a last-in-first-out fashion. A platoon of appropriate size breaks away from the end of the queue and backs up. This platoon will stop its backward motion when it creates sufficient spacing between the front vehicle and the last vehicle of the remaining queue. The backup distance depends on the speed of the adjacent lanes and the constraints on acceleration and jerk. The platoon then accelerates to the speed of an adjacent lane and changes lane whenever an appropriate gap ap-

proaches. To facilitate the process, the link layer can order the creation of gaps in the adjacent lanes upstream of the incident. It should be noted that the initiation of the *Queue Management* maneuver and its efficient operation rely on cooperation from the link layer. Like all things depending on the link layer, this maneuver is not safety critical and can be abandoned if necessary.

C. Regulation Layer

The normal mode regulation supervisor originally designed in [33], is a finite-state machine whose transitions depend upon the commands from the coordination layer, the readings of the sensors and the state of the continuous controllers. The degraded mode regulation supervisor will be required to play a similar role. Most of the maneuvers described above can be carried out

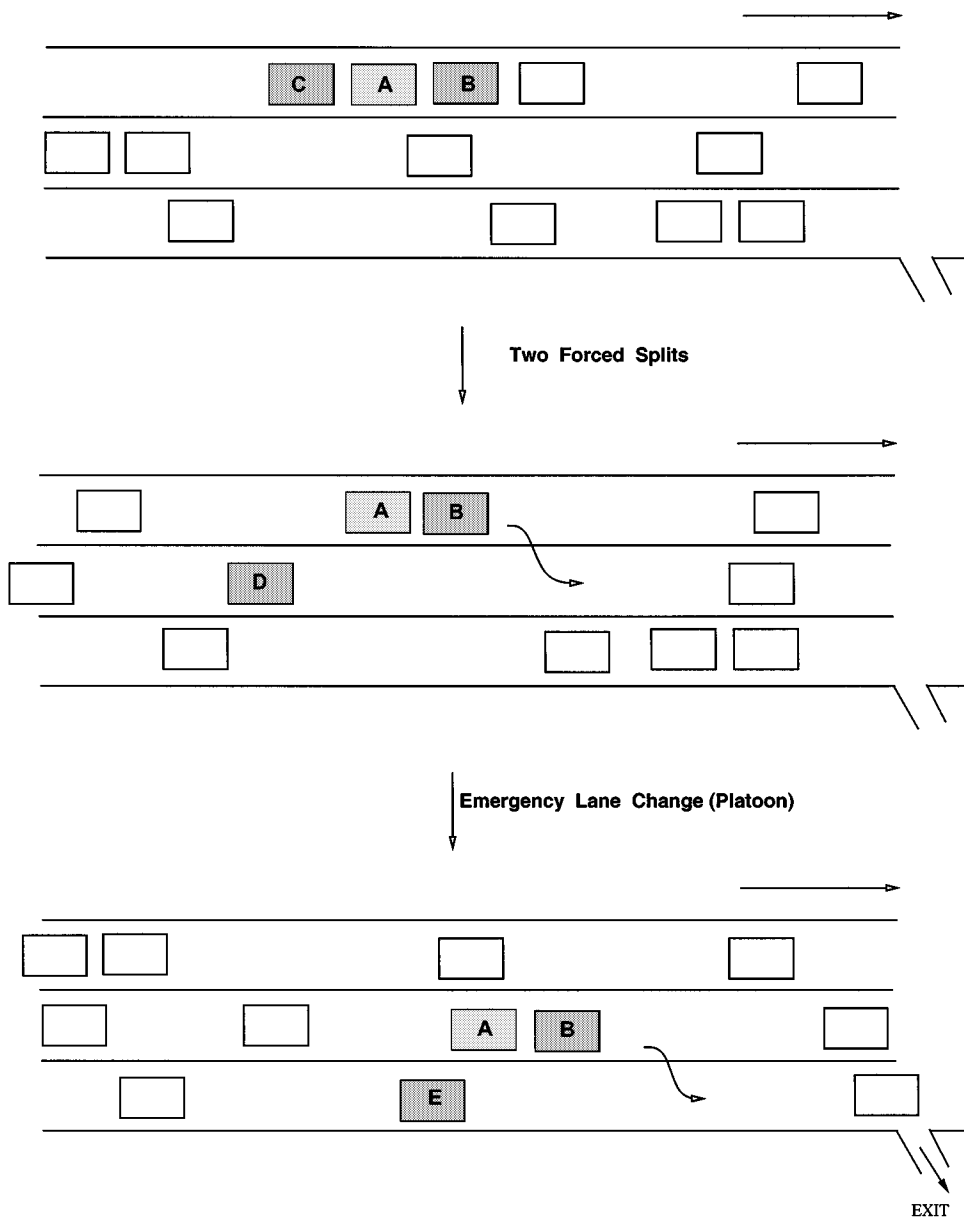


Fig. 13. Take immediate exit—escorted.

by tuning some of the regulation layer controllers designed for normal operation. For example, the maneuvers ELC and FS can use the normal mode regulation layer lane change and split controllers, respectively [24], [32]. The new maneuvers front dock and platoon lane change need separate regulation layer control laws to be designed [28], [29].

V. CONCLUDING REMARKS

We proposed a framework for an AHS design that is capable of operating in the presence of faults and other factors that induce performance degradation. Our framework is hierarchical and builds on the control hierarchy of [2]. The design provides a high degree of autonomy by extending the information structure to include data about the system capability and the control structure to make a distinction between strategic planning and execution. We sketched a design for the coordination layer

of the extended architecture and provided requirements for the link and regulation layer. Our framework has more recently been filled in with appropriate control laws for the link layer [30], [35], coordination layer [12] and regulation layer [28], [29].

A few remarks are in order concerning the design methodology proposed here.

- 1) *Design Optimality:* Any AHS controller represents a tradeoff between safety, throughput, passenger comfort, and design complexity. Quantifying these issues and determining the optimal tradeoff is an overwhelming task. No claims of optimality are made for the design presented here. The proposed controllers were derived from an intuitive understanding of what strategies are likely to be safe in a given situation. Within this set of strategies the one that allowed the vehicle move for as long as possible was selected, in an attempt to maximize throughput. The design obtained in this way is rather

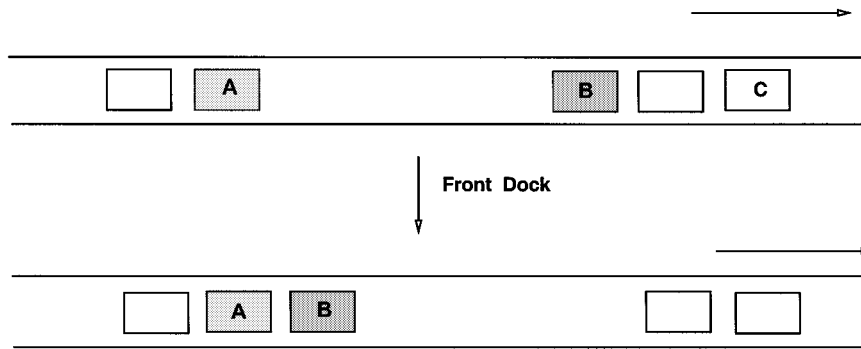


Fig. 14. Front dock maneuver.

complicated. Much simpler designs are possible; for example, the controller could stop the faulty vehicle whatever the fault and wait for an emergency vehicle to tow it out of the AHS. Intuitively such a scheme would lead to a more severe degradation in performance.

- 2) *Pseudosensors and Actuators*: A feature of the design proposed here is the use of the sensors and actuators of neighboring vehicles together with the communication devices as “pseudosensors” and “pseudoactuators” for the faulty vehicle. For example, in the *Aided Stop* strategy, a vehicle that is incapable of braking uses communication and the brakes of the vehicle ahead of it to come to a stop. Similar arrangements (for example in the TIE-E strategy) are made for vehicles that have lost sensing capabilities. This kind of interaction is an example of how communication can be used to introduce cooperation between the vehicles. While in normal mode neighboring vehicles can be viewed as malicious opponents in designing safe vehicle following and maneuver execution controllers [36], in degraded modes their actions are “controlled” by the faulty vehicle through communication. The price to pay is a substantial increase in the design complexity, and some inconvenience (and possibly delays) for the assisting vehicles.
- 3) *Verification*: After designing the control laws in this framework, the extended hierarchy has to be verified. Unfortunately there is no systematic way of verifying such complex, hybrid systems. Some aspects of the extended coordination layer control laws have been verified using computer aided verification tools [12]. The proof, however, relies on simple abstractions of the continuous dynamics of the regulation and physical layer; therefore, it will guarantee safety only if the continuous dynamics can be designed to comply with these abstractions. From past experience it is clear that designing regulation layer control laws to satisfy such abstractions can be very challenging. The examples in [37] demonstrate that even for normal operation undesirable behavior (high relative velocity collisions) can arise when combining a verified coordination layer design [19] with a set of individually satisfactory regulation layer controllers. In [36] optimal control and game theoretic methods were used to analytically verify safety properties of the normal mode hybrid dynamical system. We are hoping to extend these techniques to degraded mode verification.
- 4) *The Role of Simulation*: Because of the lack of formal analysis and design tools, simulation is likely to play an indispensable role for the evaluation of degraded mode strategies. Even though simulation can not replace formal proofs, it can still provide valuable information about the system performance. More specifically, successful results under extensive simulation indicate that the design is likely to behave well, even though there may still be room for situations where the system behaves poorly. On the other hand, unsatisfactory performance on the simulation testbed indicates design shortcomings and may suggest improvements. Finally, Monte Carlo simulation of the overall fault tolerant system can be used to obtain estimates of the impact of the degraded mode controllers on the highway throughput and hence validate any theoretical models developed for this purpose. The AHS simulator SmartPath [34] has been successfully used in the past to carry out all these tasks for the normal mode [3], [37].

ACKNOWLEDGMENT

The authors would like to thank S. Alag, L. Alvarez, A. Deshpande, J. Frankel, K. Goebel, R. Horowitz, P. Li, A. Lindsey, A. Puri, S. Sastry, E. Singh and P. Varaiya for helpful discussions providing insight into this problem. They also would like to thank S. Choi, B. Foreman, C. Gerdes, D. Maciucă, D. Koller, S. Patwardhan, and D. Swaroop for their help in compiling a list of faults and adverse environmental conditions and determining their impact on the AHS.

REFERENCES

- [1] P. Ioannou and C. Chien, “Autonomous intelligent cruise control,” *IEEE Trans. Veh. Technol.*, vol. 42, no. 4, pp. 657–672, 1993.
- [2] P. Varaiya, “Smart cars on smart roads—Problems of control,” *IEEE Trans. Automat. Contr.*, vol. 38, no. 2, pp. 195–207, 1993.
- [3] B. S. Y. Rao and P. Varaiya, “Roadside intelligence for flow control in an IVHS,” *Transport. Res.*, pt. C, vol. 2, no. 1, pp. 49–72, 1994.
- [4] V. Garg, “Fault detection in nonlinear systems: An application to automated highway systems,” Ph.D. dissertation, Dept. Mechanical Eng., Univ. California, Berkeley, 1995.
- [5] M. Tomizuka, S. Patwardhan, W. B. Zhang, and P. Devlin, “Theory and experiments of tire blow-out effects and hazard reduction control for automated vehicle lateral control system,” in *Proc. Amer. Contr. Conf.*, 1994, pp. 1207–1209.
- [6] A. Hitchcock, “A specification of an automated freeway with vehicle borne intelligence,” Inst. Transport. Studies, Univ. California, Berkeley, Tech. Rep. PATH Memo. 92-8, 1992.

- [7] J. E. White and J. L. Speyer, "Detection filter design: Spectral theory and algorithms," *IEEE Trans. Automat. Contr.*, vol. AC-32, no. 7, pp. 563–606, 1987.
- [8] R. K. Douglas, J. L. Speyer, D. L. Mingori, R. H. Chen, D. P. Malladi, and W. H. Chung, "Fault detection and identification with application to advanced vehicle control systems," Inst. Transport. Studies, Univ. California, Berkeley, California PATH Res. Rep. UCB-ITS-PRR-95-26, 1995.
- [9] A. Agogino, K. Gobel, and S. Alag, "Intelligent sensor validation and sensor fusion for reliability and safety enhancement in vehicle control," Inst. Transport. Studies, Univ. California, Berkeley, California PATH Res. Rep. UCB-ITS-PRR-95-40, 1995.
- [10] M. Sampath, R. Sengupta, S. Lafortune, and K. Sinamohideen, "Diagnosability of discrete-event systems," *IEEE Trans. Automat. Contr.*, vol. AC-40, no. 9, pp. 1555–1575, 1995.
- [11] R. Sengupta, "Diagnosis and communication in distributed systems," in *Wkshp. Discrete-Event Syst.*, Cagliari, Italy, 1998.
- [12] D. N. Godbole, J. Lygeros, E. Singh, A. Deshpande, and A. Lindsey, "Communication protocols for a fault tolerant automated highway system," *IEEE Trans. Contr. Syst. Technol.*, to be published.
- [13] H.-S. Tan, R. Rajamani, and W.-B. Zhang, "Demonstration of an automated highway platoon system," in *Proc. Amer. Contr. Conf.*, 1998, pp. 1823–1827.
- [14] J. B. Michael, D. N. Godbole, J. Lygeros, and R. Sengupta, "Capacity analysis of traffic flow over a single-lane automated highway system," *ITS J.*, vol. 4, no. 1, pp. 49–80, 1998.
- [15] J. Carbaugh, D. N. Godbole, and R. Sengupta, "Safety and capacity analysis of automated and manual highway systems," *Transport. Res.*, pt. C, vol. 6, pp. 69–99, 1998.
- [16] P. Li, R. Horowitz, L. Alvarez, J. Frankel, and A. Robertson, "An automated highway system link layer controller for traffic flow stabilization," *Transport. Res.*, pt. C, vol. 5, no. 1, pp. 11–37, 1997.
- [17] R. W. Hall, "Longitudinal and lateral throughput on an idealized highway," *Transport. Sci.*, vol. 29, no. 2, pp. 118–127, 1995.
- [18] M. E. Broucke and P. Varaiya, "A theory of traffic flow in automated highway systems," *Transport. Res.*, pt. C, vol. 4C, no. 4, pp. 181–210, 1996.
- [19] A. Hsu, F. Eskafi, S. Sachs, and P. Varaiya, "Protocol design for an automated highway system," *Discrete-Event Dynamic Syst.*, vol. 2, no. 1, pp. 183–206, 1994.
- [20] D. N. Godbole, F. Eskafi, E. Singh, and P. Varaiya, "Design of entry and exit maneuvers for AHS," in *Proc. Amer. Contr. Conf.*, 1995, pp. 3576–3580.
- [21] J. K. Hedrick, D. McMahon, V. Narendran, and D. Swaroop, "Longitudinal vehicle controller design for IVHS system," in *Proc. Amer. Contr. Conf.*, 1991, pp. 3107–3112.
- [22] H. Peng and M. Tomizuka, "Vehicle lateral control for highway automation," in *Proc. Amer. Contr. Conf.*, 1990, pp. 788–794.
- [23] S. Sheikholeslam and C. A. Desoer, "Longitudinal control of a platoon of vehicles," in *Proc. Amer. Contr. Conf.*, 1990, pp. 291–297.
- [24] D. N. Godbole and J. Lygeros, "Longitudinal control of the lead car of a platoon," *IEEE Trans. Veh. Technol.*, vol. 43, no. 4, pp. 1125–1135, 1994.
- [25] W. Chee and M. Tomizuka, "Lane change maneuver of automobiles for the intelligent vehicle and highway systems (IVHS)," in *Proc. Amer. Contr. Conf.*, 1994, pp. 3586–3587.
- [26] S. Shladover, "Nat. Automated Highway Syst. Consortium Milestone 2 Rep.: Downselect Syst. configurations," Detroit, NAHSC Rep., 1997.
- [27] J. Lygeros, D. N. Godbole, and M. E. Broucke, "Toward a fault tolerant AHS design—Part I: Extended architecture," Inst. Transport. Studies, Univ. California, Berkeley, Tech. Rep. UCB-ITS-PRR-96-14, 1996.
- [28] J. Carbaugh, "Smartpath regulation layer implementation: A user's guide," Inst. Transport. Studies, Univ. California, Berkeley, PATH Res. Rep., UCB-ITS-PRR-97-48, 1997.
- [29] C. Toy, K. Leung, L. Alvarez, and R. Horowitz, "Emergency vehicles maneuvers and control laws for automated highway systems," in *Proc. 1998 ASME IMECE*, 1998.
- [30] C. Toy, L. Alvarez, and R. Horowitz, "A traffic flow controller for non-stationary velocity profiles on automated highway," in *Proc. 1999 ACC*, 1999.
- [31] D. Swaroop, "String stability of interconnected systems: An application to platooning in automated highway systems," Ph.D. dissertation, Dept. Mechanical Eng., Univ. California, Berkeley, 1994.
- [32] P. Li, L. Alvarez, and R. Horowitz, "AHS safe control laws for platoon leaders," *IEEE Trans. Contr. Syst. Technol.*, vol. 5, no. 6, pp. 614–628, 1997.

- [33] J. Lygeros and D. Godbole, "An interface between continuous- and discrete-event controllers for vehicle automation," *IEEE Trans. Veh. Technol.*, vol. 46, no. 1, pp. 229–241, 1997.
- [34] F. Eskafi, D. Khorramabadi, and P. Varaiya, "An automated highway system simulator," *Transport. Res.*, pt. C, vol. 3, no. 1, pp. 1–17, 1995.
- [35] L. Alvarez, R. Horowitz, S. Chao, and G. Gomes, "Optimal desired traffic flow patterns for automated highway systems," in *Proc. Amer. Contr. Conf.*, 1998, pp. 1828–1832.
- [36] J. Lygeros, D. N. Godbole, and S. Sastry, "Verified hybrid controllers for automated vehicles," *IEEE Trans. Automat. Contr.*, vol. 43, no. 4, pp. 522–539, 1998.
- [37] D. N. Godbole, J. Lygeros, and S. Sastry, "Hierarchical hybrid control: An IVHS case study," in *Hybrid Systems II*, P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, Eds. New York: Springer-Verlag, 1995, pp. 166–190.



John Lygeros (S'91–M'97) received the B.Eng. degree in electrical and electronic engineering in 1990 and the M.Sc. degree in control and systems in 1991, both from Imperial College of Science Technology and Medicine, London, U.K. In May 1996, he received the Ph.D. degree at the Electrical Engineering and Computer Sciences Department of the University of California, Berkeley.

From June to October 1996 he was a visiting Postdoctoral Researcher with the National Automated Highway Systems Consortium, at the Institute of Transportation Studies, University of California, Berkeley. Between November 1996 and September 1997 he was a Postdoctoral Research Associate with the Laboratory for Computer Science at Massachusetts Institute of Technology. He is currently a Postdoctoral Researcher at the Electrical Engineering and Computer Sciences Department of University of California, Berkeley, and holds a part time Research Engineer position at SRI International. His research interests include hierarchical and hybrid systems, nonlinear control theory and their applications to highway systems and air traffic management.

Dr. Lygeros is the corecipient of the 1997 Elisha Jury Award for "Excellence in Systems Research," from the Electrical Engineering and Computer Sciences Department of the University of California, Berkeley.



Datta N. Godbole (S'93–M'95) received the B.E. degree in electrical engineering from the University of Pune, India, in 1987, the M.Tech. degree in systems and control engineering from the Indian Institute of Technology, Bombay, India, in 1989, and the Ph.D. degree in electrical engineering and computer sciences from the University of California, Berkeley, in 1994.

From January 1995 to January 1999, he worked as an Assistant Research Engineer with the California PATH program, Institute of Transportation Studies, University of California, Berkeley. Since February 1999, he is working as a research scientist at Honeywell Technology Center, Minneapolis, Minnesota. His research interests include hybrid control systems, hierarchical control of complex multiagent systems, nonlinear control, and applications to intelligent vehicle highway systems, air traffic management systems, flight control, and control of space vehicles.

Dr. Godbole is a recipient of the 1987 University Gold Medal in engineering from the University of Pune, India, and a corecipient of the 1997 Elisha Jury Award for "Excellence in Systems Research" from the Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley.

Mireille Broucke received the B.S.E.E. degree from the University of Texas, Austin, in 1984, the M.S.E.E. degree from the University of California, Berkeley, in 1987, and since 1996 has been pursuing the Ph.D. degree in EECS at the University of California, Berkeley.

She has ten years experience in the aerospace and transportation industries. From 1988 to 1993 she was with Integrated Systems developing software tools for nonlinear systems analysis. From 1993 to 1996 she was with the California PATH program as a program manager and researcher. Her research interests include system theory, especially geometric methods in control, and hybrid and hierarchical systems.