AUTONOMY IN A RESTRICTED WORLD

Ümit Özgüner, Cem Hatipoğlu and Keith Redmill

The Ohio State University, Department of Electrical Engineering, 2015 Neil Ave.

Columbus, OH 43210, USA

Keywords: Hybrid systems, autonomy, road vehicles.

ABSTRACT

This paper studies "autonomy" in the context of Automated Highway Systems (AHS) and explores means of designing hybrid controllers for local decision making in an autonomous vehicle. A supervisory controller is developed which generates event-driven commands to the low-level continuous state controllers. The presented work is an overview of the top level controller for OSU Demo '97 (autonomous) vehicles.

INTRODUCTION

Autonomy is loosely defined as the ability to make decisions based on local measurements, analysis and a local value system.

In the context of an Automated Highway System, autonomy implies local decision making in a vehicle without commands or coordination signals from the infrastructure, or real-time information actively and deliberately transmitted from other moving vehicles.

Local decision making in a vehicle could be at a low level such as steering, or at a high level, such as "pass initiation", or at an even higher level, such as "exit highway". A vehicle (or any system) could have autonomy for low level decision making, but not be autonomous at a higher level. Thus a vehicle could be autonomous for steering, throttle control and brakes, but still get commands for general route guidance.

Autonomy also implies having access to a local "world model". Measurements, such as speed, distance to car in front, direction of road leading away, are all pieces of information that an "intelligent vehicle" uses to build its world model. Signals returned from a radar reflective stripe, or a magnet embedded into the asphalt also provide sensory information about the world, unless these are deliberately coded to direct the car in its operations. Thus getting information from infrastructure does not necessarily imply loss of autonomy.

There are two reasons why we are interested in autonomy issues in cars. First, we would prefer to have the cars make their own decisions without massive intervention from external sources (computers, communication networks, etc.) Existence of such an active infrastructure poses additional problems ranging from latency, to reliability to even stability. Second, although never explicitly stated, we want our automated cars to emulate (to a certain extent) human driving behavior, preferably our own.

As a decision maker, a car can base its decisions on a number of different variables. The decisions can be "event driven" (if "deer jumps onto road" then "apply maximum break pressure"), or "time driven" (if "following slow driver for 12 minutes" then "lane change initiate"), or based on continuous situation assessment ("car in front slower than our set speed" then "slow down to match speed according to given profile").

The question that needs to be answered is whether an autonomous entity with given measurement and decision making capability can "survive" in a freewheeling world. At the

0-7803-4269-0/97/$10.00 © 1998 IEEE

limit, this asks the question about a totally autonomous car moving in free traffic with all the manually driven cars.

To be able to even approach the above question we have to deal with much simpler situations first. In the present paper we will consider a much simpler world, a world with one or two unidirectional lanes and very restricted personalities. It is expected that such building blocks can be used for more complex analysis eventually.

THE WORLD AND PERSONALITIES

Consider first a "world" comprised of a single lane. The decision making here is quite simple. Any car, with a specified specified speed and headway will just need to check for a car ahead. Once following the car ahead, we can consider a number of different reactions if it changes speed.

We shall now introduce the concept of "personality" to represent a set of behavior patterns. This set will affect the decisions of the car. At this initial stage we shall consider four classes:

1. Grandma
2. Teenager
3. Jack
4. The homicidal driver

With the single lane world constraint, the first three personalities do not provide much variation. The teenager may be more aggressive in its pursuit of the car ahead. Other combinations can easily be analyzed. With many cars on the single lane, it is obvious that eventually all traffic would accumulate behind grandmas. Unless there is a homicidal Driver. Assuming such a driver could reverse direction, creates many unmanageable situations.

When we expand our world to two lanes and allow passing, even with the first three personalities, a number of interesting situations arise.

We will make a set of assumptions about the personalities.

- Grandma drives slow. Never passes.
- Teenager follows slow car ahead, but has two specific characteristics that distinguish her from Jack: A short "boredom time", a low threshold risk analysis. (Teenager will change lanes with a narrow merge distance.)
- Each car can show variations in its driving pattern, staying within its personality.

The last item above allows us to consider manual drivers together with automated vehicles. Indeed, we have assumed a manual Grandma and automated Teenager and Jack, during our runs in the 1997 Technology Demonstration. It was assumed that Grandma could speed up, slow down and even stop. It would not do a sharp stop, change lanes, or speed more than a limit (less than Teenager and Jack). The analysis of the scenario and the hybrid system model is provided in the next section.

OSU DEMO SCENARIO

The scenario involves three cars, two of which are fully automated and the other is manually driven. The demo took place on a two-lane, unidirectional, 7.6 mile long road segment in Southern California. The vehicles in the scenario are referred as the "Grandma" (manually driven car), the "Teenager" (automated car #1) and "Jack" (Automated car #2). The OSU Demo '97 scenario is described in detail in [3]. Additionally, we state that, as far as the controllers, sensors and actuators are concerned, all automated cars (the two in the demo, and the spare) are identical. Hence, their roles, in the demo, are interchangeable upon modifying a couple of parameters which describe the "personality" of the vehicle.

Each automated vehicle is capable of performing headway (slow down, speed up, cruise, stop) and lateral control (lane keeping, lane change). The task allocation details will be explained next.

THE FUNCTIONAL ARCHITECTURE

Functional hierarchies arrange decision making and problem solving processes as a hierarchical

tree. In this tree, goals are decomposed into subgoals such that accomplishments of these subgoals contribute to the accomplishment of the original goal. Following the proposed approach, we assume the complete system is given with the knowledge about its functional behavior. First, we decompose the system into its physical components. Then, we associate functions to the decomposed components to achieve a structurally and functionally decomposed system.

Once the nodes of the hierarchy are determined, the next step is to connect these nodes in a coordinated way. Then, we consider a fraction of the hierarchical structures from the set of all, such that the structural coordinability axioms are satisfied. To represent the functions of the nodes and to control the system, we introduce six primitives. These primitives exist at every node of the hierarchy and their descriptions are relative to the node to which they belong. We list the primitives and the brief descriptions below with respect to any node.

Goals are assertions representing the end result to be obtained at the node.

Tasks are elementary job descriptions at the node.

Procedures are methods of accomplishing tasks. These procedures are separated into two groups:

1. *Current Procedures* are those procedures which are currently applicable at the node, and
2. *Sub-procedures* are those procedures which are applicable by sub-nodes.

Measurements are the data available from the sensors.

Constraints are task restrictions or exemptions.

1. *System Dependent Constraint*, are constraint which depend solely on the system and do not change with the environment or the goals,
2. *System Independent Constraints* are constraints which do not depend on the system, but on the environment or on the assigned goals.

Resources are task restrictions or exemptions which are related to the use of procedures.

HYBRID MODELING

It is clear that a "model" is essential for system analysis and controller design purposes. Any physical hybrid system behavior evolves as a result of a continuous time system interacting with the operation of a discrete event system (DES). In this paper, we specify the DES portion of the hybrid system to be a finite state machine (finite automaton). Note that, we are dealing with large scale hybrid systems (namely the three vehicle demo scenario) where the individual "units" (i.e. an individual autonomous vehicle) of this so-called multi-unit system are hybrid systems themselves. They interact with each other at the DES and CSS levels and the evolution of their states depend on the rest of the units (within the big picture) as well as their own since they are linked to one another through various on-board sensors.

Continuous State System Model

The continuous state dynamics of an individual autonomous vehicle is composed of continuous state space differential equations. They describe the motion of the vehicle for a given set of input-output pairs. The complexity of a vehicle model can vary significantly based on the application needs although generally the simplest valid model is picked for controller design purposes. Various low-level controllers are designed locally in the CSS such as a master speed controller (MSpC) and a master steering controller (MStC). MSpC and MStC assume inputs from the local continuous state system only, therefore can be treated as an integral part of the CSS if kept in the loop once and for all times during the autonomous operation. In this context, the

CSS system is capable of performing the following:

- Automated stop, speed up and slow down,
- Automated lane keeping and lane change using radar and/or vision based steering data.

The control problem at the continuous level is to generate reference signals to the MSpC and MStC to have the vehicle perform desired maneuvers. The DES generates the event driven commands to the CCS which is going to be explained next.

Discrete Event System Model

The discrete event system portion of the hybrid system is modeled as a finite state machine whose state transitions are manipulated by some "events" that occur in the CSS. These events are processed in the CSS (with filters and observers) and are passed to the DES side through an interface.

Let the set of low level events and requests be denoted by the set $\mathcal{E}_c$. In fact, $\mathcal{E}_c$=[$\mathcal{E}_{c_1}$, $\mathcal{E}_{c_2}$, $\mathcal{E}_{c_3}$, $\mathcal{E}_{c_4}$, $\mathcal{E}_{c_5}$, $\mathcal{E}_{c_6}$]^T where the event sets $\mathcal{E}_{c_1}$, $\mathcal{E}_{c_2}$, $\mathcal{E}_{c_3}$, $\mathcal{E}_{c_4}$, $\mathcal{E}_{c_5}$ and $\mathcal{E}_{c_6}$ are given by $\mathcal{E}_{c_1} = \{e_1^1, e_1^2\}$, $\mathcal{E}_{c_2} = \{e_2^1, e_2^2\}$, $\mathcal{E}_{c_3} = \{e_3^1, e_3^2\}$, $\mathcal{E}_{c_4} = \{e_4^1\}$, $\mathcal{E}_{c_5} = \{e_5^1, e_5^2\}$ and $\mathcal{E}_{c_6} = \{e_6^1, e_6^2\}$ which describe the events as summarized in Table 1. Those events and requests are interpreted at the interface layer and passed to the DES side.

Table 1: The list of low level events and requests.

e_1^1	:	Manual Request;
e_1^2	:	Try Auto Request;
e_2^1	:	Start Request;
e_2^2	:	Stop Request;
e_3^1	:	Left Lane Change Request;
e_3^2	:	Right Lane Change Request;
e_4^1	:	Auto Zero Request;
e_5^1	:	Target Acquired Ahead;
e_5^2	:	No target acquired ahead;
e_6^1	:	Ready for Lane Change;
e_6^2	:	Vehicle Following Mode;

There is one discrete state variable $X_i \in \mathcal{D}_1$. See Table 2 for a list of those states.

Table 2: Top level discrete states.

Manual State
Auto Stationary State
Go State
Acquire State
Follow State
Ready Left State
Lane Change Left State
Pass State
Ready Right State
Lane Change Right State
Continue State
Stop State

Table 3: Lateral DES states.

Lane Keep State
Lane Change Start State
Lane Change Complete State
Fail to Manual State

Table 4: Longitudinal DES states.

Starting State
Stop State
Pre Slowing State
Slowing State
Pre Acceleration State
Acceleration State
Pre Cruising State
Cruising State
Pre Following State
Following State

The discrete system state transition function $F(\cdot,\cdot)$ is defined as,

$$X_i(k+1) = F\{X_i(k), \phi(z(kT))\} \qquad (1)$$

where $\phi(z(kT)) \in \mathcal{E}_c$ denotes the low level events passed on to the DES side defined by the function $F : \mathcal{D}_1 \times \mathcal{E}_c \rightarrow \mathcal{D}_1$ which is not presented here due to page limitations.

Interface Layer

Recall that the objective of the controller design is to develop a DES supervisor to regulate the system dynamics to the desired trajectories. The interface layer has the major task of building the bridge between the continuous state system and the discrete state system (within the hybrid model) as shown on Figure 1. the DSS is a discretized version of the CSS. The local orientation sensing devices on the vehicle provide information regarding the vehicle's position with respect to the "World", where *World* is defined as the combination of the road and the other vehicles traveling on the highway in this AHS example. The sensors equipped on the car, and the information they provide to the supervisory controller can be summarized as follows: (i) *Vehicle ECUs* provide speed, engine RPM, steering angle, brake pressure, throttle opening and transmission measurements, (ii) A *Look-Ahead Radar* detects the offset from the lane center at a look-ahead point processing the return signal from the frequency selective stripe (FSS). It also measures the relative longitudinal distance to the near-most vehicle ahead, (iii) *Laser Range Finder* is capable of providing the relative distance Δd_i, relative velocity $\Delta \nu_i$, the relative angle to the target $\Delta \gamma_i$ and the width of the target. It can track multiple vehicles (identified with subscript i) at a time. (iv) A *CCD Camera* processes the picture frames to detect the markers on either (or both) side(s) of the lane and outputs a similar (to the look-ahead radar output) offset signal, (v) A *Timer* is used to measure how long the controlled vehicle stays at an operational DES state, (vi) *Side-Looking Radars* provide the information whether or not there exists another vehicle in the adjacent (possibly passing) lane. (vii) A *Yaw Rate Sensor* is used mainly in the low level controllers within the CSS system, however, it also provides redundant information regarding the vehicle orientation which is passed to the DES side.

The DES supervisor will command some CSS inputs through the interface unit (D/A converters) and the controller design procedure will be discussed shortly. When the CSS and DSS were presented in the previous sections, it has been indicated that there needs to be a two-way information flow through the interface. We define the interface from the continuous time system to the DSS by (the partial function),

$$\phi : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathcal{E}_c \tag{2}$$

so that $\phi(x(t), u(t)) = e_p^s(t)$ represents a low level event in $\mathcal{E}_c$ that occurs at the time instant t, where n and m are the orders of the continuous state and input spaces respectively. "$u(t)$" denotes the control generated by the CSS, which refers to the low level controller inputs like the throttle opening, brake pressure, steering signal, e.t.c.

The supervised control law generated by the DES supervisor is interpreted at the Interface layer for information flow from DES to CCS, and is passed to the CSS input generator. Based on the suggested control law, certain continuous input is selected from a finite set of all possible inputs to fully automate the motions of the considered vehicle. The interface to the continuous time system, which specifies how logical operations in the system can directly influence the continuous time system is given by (the partial function),

$$\psi : \mathcal{P}(\mathcal{E}) \times \mathcal{X} \rightarrow \mathbb{R}^l \tag{3}$$

where $\mathcal{P}(\mathcal{E})$ denotes the power set of the events set, X is the set of DES states and l is the order of the continuous inputs generated through the DES supervisor. A sample designed controller can be found in [1].

CONCLUSIONS

We have particularly focused on a three vehicle demonstration scenario as an application example. An AHS is a multi-lane world with many vehicles operating on it. It can be modeled as a multi-agent, large-scale hybrid system with the control objectives being specified accordingly in a decentralized manner. In this paper, we have dealt with a relatively easier case which involves three vehicles operating on a two lane highway.

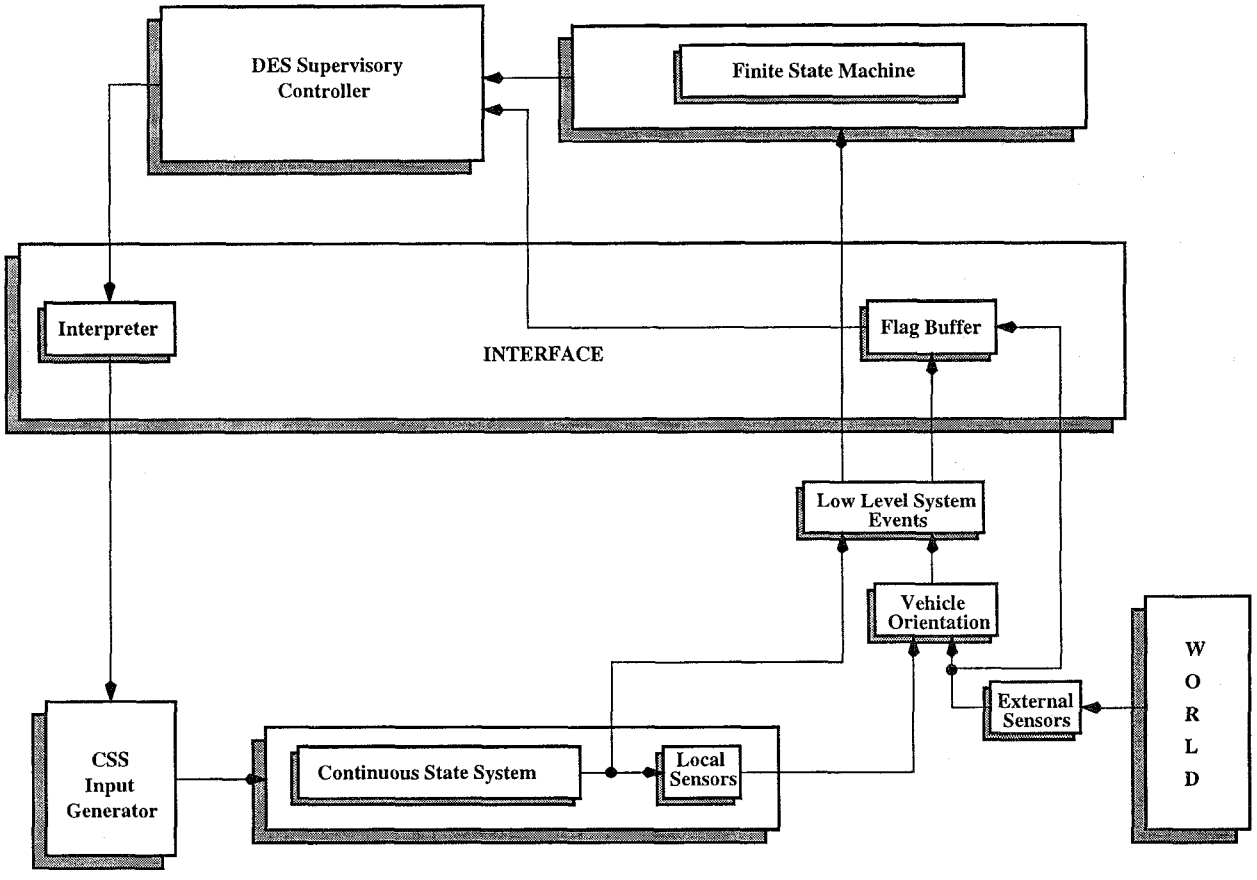

Figure 1: DES, CSS and Interface Scheme

On the other hand, the studied three vehicle scenario is considered as a step towards a more generalized case where (arbitrarily) many vehicles exist on a highway with (arbitrarily) many lanes. Let us refer to this as a "generalized AHS". Considering the "simplified AHS" we have examined in this paper, and comparing it to the "generalized AHS", we immediately observe the need for a different sort of decomposition, because (i) Lane change maneuvers are no longer one dimensional (in general) and (ii) More than 2 external vehicles might effect the supervised decision. From the controlled vehicle's point of view, the dominant units are the closest vehicles in the same lane ahead and the ones on the two adjacent lanes. Clearly, there will be more number of states in the finite-state machine as there are more low-level events coming from the continuous state system.

However, the results of this paper's work can be modified relatively easily to fit the structure and needs of the so-called "generalized AHS".

References

[1] Ü. Özgüner, C. Hatipoğlu, A. İftar and K. Redmill, "Hybrid Control Design for a Three Vehicle Scenario Demonstration using Overlapping Decompositions", to appear in *Hybrid Systems IV.*

[2] J. Lygeros, D.N. Godbole and S. Sastry, "A Verified Hybrid Controller for Automated Vehicles", Proceedings of the 1996 Conference on Control and Decision, Kobe, Japan, pp 2289-2294 , 1996.

[3] Ü. Özgüner, B. Baertlein, C. Cavello, D. Farkas, C. Hatipoğlu, S. Lytle, J. Martin, F. Paynter, K. Redmill, S. Schneider, E. Walton, J. Young "The OSU Demo '97 Vehicle", IEEE Conference on Intelligent Transportation Systems, Boston, MA, November 1997.