AN AUTONOMOUS DRIVER MODEL FOR THE OVERTAKING MANEUVER FOR USE IN MICROSCOPIC TRAFFIC SIMULATION

OMAR AHMAD
oahmad@nads-sc.uiowa.edu

YIANNIS E. PAPELIS
yiannis@nads-sc.uiowa.edu

National Advanced Driving Simulator & Simulation Center
The University of Iowa
http://www.nads-sc.uiowa.edu

ABSTRACT

Autonomous driver models are widely used in driving simulators as an integral part of microscopic traffic simulation within the virtual environment. Realistic and believable driver models significantly enhance the user's experience and are often necessary for conducting simulator-based research. One way to increase the traffic fidelity is to extend the behaviors exhibited by driver models. Typically, behaviors such as vehicle following and lane tracking are augmented with specialized behaviors such as emergency responses and parking maneuvers. This paper describes one such behavior that is part of the microscopic traffic simulation system developed at the University of Iowa and currently in use in various driving simulators. The paper describes the model used for overtaking a vehicle by temporarily traveling in an opposite-flow lane.

INTRODUCTION

In a driving simulator environment, the driver's experience is based largely upon interaction with surrounding traffic. Providing a realistic simulation of individual traffic elements, often referred to as a microscopic traffic simulation, is a necessary component of high-fidelity driving simulation.

Much of the work in autonomous driver modeling can, to some degree, be applied to driving simulator applications. In general, this work focuses on the development and validation of models that cover the overall spectrum of behaviors exhibited by drivers [1], such as vehicle following or lane tracking. Specialized driver models have been developed for various traffic analysis applications, including the effect of changes to the road infrastructure on traffic density [2] and the optimal design of tollway plazas [3]. Such models are limited when applied to driving simulation because they are not comprehensive; i.e., they do not address all situations that can be encountered in the virtual environment. For example, if the virtual environment includes controlled intersections, a lane-tracking model, no matter how accurate, will provide unacceptable results if it simply drives through intersections without yielding.

This paper focuses on a specialized behavior that is part of a comprehensive driver model: overtaking a vehicle by temporarily traveling in an opposite-flow lane. Overtaking, also known as passing, occurs often on rural roads where it is the only way to get around a slow-moving vehicle. It is an interesting behavior to model because, although it appears simple, it involves several decision-making points and continuous re-evaluation of the situation.

The remainder of this paper is organized as follows. First, a brief description of the formalism used to implement the overall driver model helps to interpret the design of the passing behavior. Next, a short description of the overall driver model details the framework within which the passing behavior operates. A detailed description of the passing behavior model is provided next, followed by a short conclusion.

HCSM FORMALISM

The autonomous vehicle model has been implemented using Hierarchical Concurrent State Machines (HCSMs) [4]. HCSMs extend the traditional state machine formalism by introducing hierarchy and concurrency. Hierarchy is introduced by allowing an HCSM to have zero or more child HCSMs. We collectively refer to a top-level HCSM and all of its children as the HCSM tree. Execution of the model takes place by executing two pieces of code associated with each HCSM in the tree: a pre-activity and a post-activity. Executing an HCSM is an in-order traversal of the HCSM tree. During the first visit to the HCSM, the pre-activity executes and the children are visited. During the second visit, the post-activity executes. Concurrency is introduced by allowing an HCSM to have either concurrent or sequential children. When concurrent, all children are visited and their activities executed. When sequential, only one child is visited. The children of a sequential HCSM are connected with transitions. As in traditional state machines, each transition is associated

with a predicate function that executes every frame. When the predicate evaluates to true, the transition fires and control is transferred from the originating to the terminating HCSM.

Several methods facilitate inter- and intra-HCSM communication, but only the input and output are necessary to describe the model for the passing behavior. Each HCSM can have any number of input and output parameters that can only be accessed between HCSMs that have a parent-child relationship. When building a model, a parent HCSM can send information to its child's input parameters and read information from its child's output parameters. Typically, a parent HCSM writes values to its child's inputs during its pre-activity function and reads values from its child's outputs during its post-activity function. Both input and output parameters can be marked as having "no value."

AUTONOMOUS DRIVER MODEL FRAMEWORK

Figure 1 illustrates the architecture of the overall traffic simulation framework. It displays two major components. The HCSM Runtime System maintains the instances of the HCSM trees and executes the HCSMs. The Correlated Virtual Environment Database (CVED) maintains information about the physical environment and provides flexible queries that provide real-time access to information about the environment.

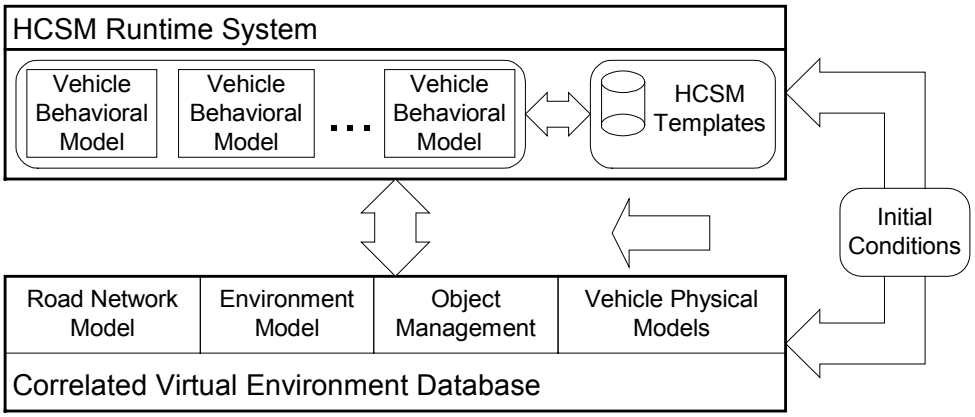

Figure 1:
Overall driver model framework.

The HCSM Runtime System initially starts with only a list of HCSM templates that represent HCSM trees that can be instanced. A set of initial conditions, shown outside the HCSM system block, specifies one or more templates that will be instanced during initialization. Generally, they represent higher level managers whose job is to dynamically instance additional HCSMs representing the various traffic participants. During instancing, an arbitrary number of parameters can be specified to an HCSM, thus providing traffic variability. Each HCSM instance represents an instance of one autonomous driver model.

CVED provides the real-time interface to the virtual environment. It provides a rich set of functions to obtain and set the state of the virtual environment. The virtual environment comprises several subcomponents, some static and some dynamic. The road network model, which includes the road pavement description, road markings, intersections, etc., is

a static component that cannot be modified at runtime. Dynamic components include environment conditions such as wind and rain, and all objects, including vehicles, traffic signs, traffic lights, etc. Note that CVED vehicles represent the physical presence of a traffic element, while HCSMs represent the behavioral model of a traffic element. There is a one-to-one correspondence between an HCSM instance representing a driver model and a CVED object representing the vehicle controlled by the driver model. Furthermore, each CVED vehicle object is associated with a dynamic model that is responsible for calculating the motion of the vehicle based on control inputs provided by the associated HCSM.

An external synchronization agent controls execution of the system. It calls functions that execute all HCSMs and then calls functions that execute the various physical models. One of the advantages of decoupling the behavioral and physical models is that they can execute in different frequencies. For the remainder of this paper, we use the term *frame* to refer to the execution of all HCSM trees followed by execution of the physical models.

To better understand the operation of the system, consider a typical sequence of events that takes place upon instancing of an HCSM. For simplicity, we will refer to the HCSM as "executing" as a whole without specifying the internal details. The system starts by executing an iteration of all HCSMs. The appropriate manager HCSM executes and instances our example HCSM. Upon creation, the example HCSM calls CVED functions that create a new CVED object. The initial placement of the object is conveyed by the HCSM, which has read it from its initial parameters. After the object is created, the HCSM uses CVED to obtain information about the surroundings, determines the appropriate speed based on the posted speed limit and factors such as road curvature, and determines a direction of travel based on the lane it was placed in. The HCSM then uses CVED to obtain a guide point located ahead of its current position. It then calculates an appropriate steering angle and a desired acceleration and provides these inputs to CVED. Once the CVED execution step takes place, the dynamic model executes one or more iterations using the control inputs associated with the object. As a result of the motion generated by the dynamics model, the object moves along the lane direction. In the next frame, the HCSM interrogates CVED to determine where the object is located, then uses this position to gather detailed data about the speed limit, direction of travel, etc. During this phase, the HCSM can query CVED to obtain any information required by the various behaviors. For example, the HCSM can query CVED to determine if another vehicle is located along the travel path, in which case a driver-follower algorithm is engaged. To implement intersection navigation, the HCSM asks if there are upcoming intersections and, if so, queries the intersection corridors and obtains any controlling signs or traffic lights. Independent of the complexity of the behavioral model, the final outcome of the HCSM's execution is a fresh set of control inputs that are provided to CVED, which in turn uses them for the dynamics model. This cycle repeats for the duration of the simulation.

GENERAL STRUCTURE OF THE OVERALL DRIVER MODEL

A detailed description of the overall driver model is beyond the scope of this paper, but some information on its structure is necessary in order to describe the overtaking behavior. Papelis and Ahmad [5] provide a comprehensive description of the overall model.

Figure 2 shows a simplified version of the overall driver model. Only two levels of the tree are shown, and the connections between the parameters of the parent and children HCSM have been eliminated to reduce clatter. The Autonomous HCSM is the root of the HCSM tree; its pre- and post-activity functions are responsible for most of the interaction between the behavioral model and CVED. Note that the remaining HCSMs in Figure 2 are children of Autonomous. All children have one input and two output parameters. The input parameter receives a composite data structure with information about the environment. Unlike CVED, which provides information about any place in the database, this data is filtered and reflects only data relevant to the actual position of the object. The pre-activity of the Autonomous HCSM queries CVED and packages the information. Of the two output parameters, one contains a speed control command, and the other contains a position guidance command. The output parameters of all children are accessible by the parent.

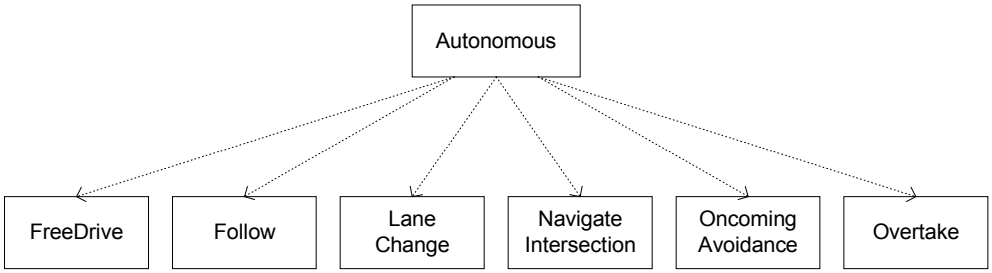

Figure 2:
Autonomous vehicle HCSM.

Each child of Autonomous is responsible for a specific behavior. FreeDrive provides position guidance and speed control based on road geometry. Follow controls speed when following another vehicle. LaneChange implements lane changes across lanes of the same direction. NavigateIntersection, a fairly complex HCSM that is only active near intersections, implements various yielding behaviors. OncomingAvoidance avoids oncoming vehicles to prevent a head-on collision. Finally, Overtake implements passing and is the focus of this paper. When a behavior is not applicable to the current situation—for example, Follow when there is no vehicle ahead—the child HCSM sets its output parameters to "no value." It is the responsibility of the Autonomous post-activity to gather all existing control commands from all children and fuse this information to come up with a single set of control inputs that will be conveyed to CVED. When HCSMs provide complementary information, fusion simply involves combining the relevant information. When they provide contradictory controls, low-priority HCSMs are ignored and, among similar-priority HCSMs, the one with the most conservative control inputs is selected. A significant part of the overall model is encapsulated in the method used to fuse conflicting or contradictory outputs, but for the scope of this paper it is sufficient to specify that the Overtake HCSM is the highest priority HCSM. Its outputs are always used. This design puts stringent requirements on the functionality of Overtake—once it takes control, it must provide continuous position information for the whole overtake maneuver. In situations where an abort is necessary, Overtake must guide the vehicle back to a position where other

HCSMs can take over without abrupt maneuvers. For example, FreeDrive is designed to closely track a lane, assuming a bounded initial error in the lateral placement of the vehicle. If Overtake aborts while the lateral distance of the vehicle from the lane being tracked exceeds the allowable error and FreeDrive takes over, the sharp change in direction could cause the dynamic model to oscillate or even become uncontrollable.

OVERTAKE HCSM

Figure 3 illustrates the design of the Overtake HCSM. Note that, unlike Autonomous, which had children executing concurrently, Overtake has sequential children. Only one will execute during each frame, and transition firings control which one will execute during the next frame. All children have the same set of output parameters, and Overtake simply propagates to its outputs the data on the output parameter of the active child. Upon initialization, the active HCSM is None.

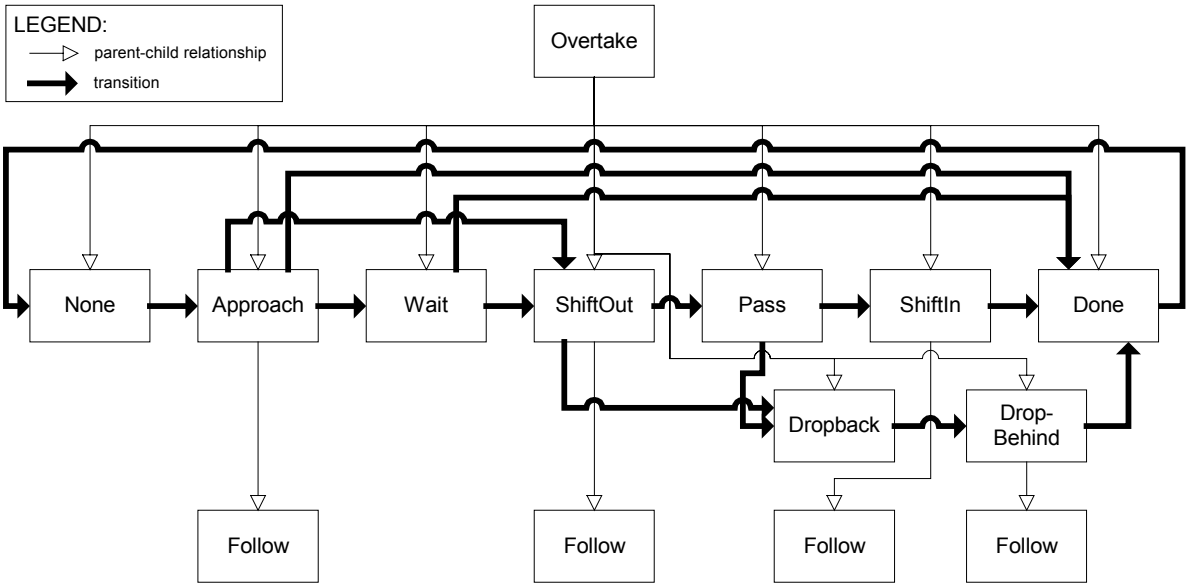

Figure 3:
Overtake HCSM.

In order to be robust, Overtake has been designed to handle dynamic changes in the environment. This involves continuous re-evaluation of the go/no-go decision because conditions that exist at the start of the maneuver may change, necessitating adaptation of the maneuver or even a complete abort. Furthermore, in order to make such decisions, it is often necessary to estimate travel times for which there is no closed-form solution. In such cases, lookup tables have been used to provide rough estimates. To accommodate small inaccuracies, margins are used when comparing actual values to acceptable thresholds. Cremer, Kearney, and Papelis [4] provide more implementation details.

NONE HCSM

None represents the initial state of the maneuver. While in the None state, no outputs are generated and no passing takes place. The predicate associated with the transition to

Approach represents the preconditions for initiating a maneuver. All preconditions have to be true to initiate the maneuver, and these preconditions include the following:
The lane to the left of the current lane has an opposite direction.
A lead vehicle exists within a certain distance, and its speed differential to the current vehicle is above a threshold.
Road signage and road markings permit passing.
The speed necessary to pass the vehicle would not violate the speed limit during the estimated maneuver duration, or would not be unachievable due to road curvature.
There is enough space to perform the passing without running into an intersection.
There is enough space between the lead vehicle and any vehicles ahead of it to accommodate the current vehicle.
The lead vehicle is not being passed by another vehicle.
No oncoming traffic would interfere for the estimated duration of the maneuver.

APPROACH HCSM

The Approach HCSM is active when the preconditions exist for starting a maneuver. It represents the phase when the passing vehicle approaches the lead vehicle but has not started shifting out of its lane. The rationale for this is that drivers want to minimize time spent in the opposite lane, so they would get near enough to the lead vehicle before shifting out. While in this state, the vehicle tracks its lane for position guidance. For velocity guidance, the state uses the Follow HCSM. It assigns a short following distance to Follow's input parameter, then looks at the velocity guidance returned by Follow and generates its own velocity of 20 percent over the speed of the lead vehicle. It then picks the more conservative of the velocities. The predicate for the transition to Approach evaluates to true when the time to collision with the lead vehicle reaches a certain threshold.

The transition to Done indicates that the conditions allowing passing are no longer true, and the vehicle aborts the maneuver. The predicate for the transition to Done is the inverse of that for the transition between None and Approach, with the exception of condition None. The predicate for the transition to Wait is oncoming traffic that would interfere with passing. The predicate from Approach to ShiftOut indicates that there is no oncoming traffic; it is, therefore, the inverse of the predicate for the transition from Approach to Wait.

WAIT HCSM

This state indicates that all conditions are amenable for passing, with the exception of oncoming traffic. While in this state, the vehicle tracks the lane and controls its speed so that the gap between it and the lead vehicle stays the same.

The transition to ShifOut indicates that there is no oncoming traffic. The predicate associated with this transition is the inverse of that for the transition from Approach to Wait. The transition from Wait to Done indicates that the conditions allowing passing are no longer true, and the vehicle aborts the maneuver. The predicate associated with this transition is same as that for the transition from Approach to Done.

SHIFTOUT HCSM

This state indicates that all conditions for passing have been met. While in this state, the vehicle controls its position to maneuver laterally from its originating lane to the oncoming lane. For velocity guidance, the state uses the Follow HCSM and assigns a short following distance to Follow's input parameter. It then looks at the velocity guidance returned by Follow and generates its own velocity guidance of 20 percent over the speed of the lead vehicle. It then picks the more conservative of the two velocities.

The transition from ShiftOut to Pass indicates that the vehicle has gained enough lateral clearance between itself and the lead vehicle to start the pass maneuver. The predicate associated with this transition evaluates to true when the distance between the vehicle's right edge and the lead vehicle's left edge exceeds some threshold. The transition from ShiftOut to DropBack indicates that the conditions allowing passing are no longer true, and the vehicle aborts the maneuver. The predicate associated with this transition is the same as that for the transition from Wait to Done.

PASS HCSM

This state indicates that the vehicle has positioned itself in the oncoming lane and has gained enough clearance to start the pass maneuver. The Pass HCSM accelerates the vehicle past the lead vehicle. In this state, the vehicle tracks the center of the oncoming lane for position guidance. For velocity guidance, it sets its own velocity of 20 percent above the velocity of the lead vehicle.

The transition from Pass to ShiftIn indicates that the vehicle has moved past the lead vehicle. The predicate associated with this transition evaluates to true once the distance between this vehicle's rear bumper and the lead vehicle’s front bumper exceeds some threshold. The transition from Pass to DropBack indicates that the conditions allowing passing are no longer true, and the vehicle aborts the maneuver. This transition's predicate is the same as that for the transition from ShiftOut to DropBack.

SHIFTIN HCSM

This state indicates that the vehicle has moved past the lead vehicle. While in this state, the vehicle controls its position to maneuver laterally from the oncoming lane to its original lane. For velocity guidance, the state uses the Follow HCSM. The transition from ShiftIn to Done indicates that the overtake maneuver is complete. This transition's predicate evaluates to true once the vehicle has reached the center of the originating lane.

DONE HCSM

This state indicates that the overtake maneuver is complete and clears any state variables associated with the overtake maneuver. The transition from Done to None indicates that the vehicle is continuing the search for opportunities to overtake other vehicles.

DROPBACK HCSM

This state indicates that conditions that initially allowed passing are no longer valid and that the vehicle is performing an abort to the overtake maneuver. For velocity guidance, this state slows the vehicle down so that it falls behind the vehicle to be overtaken. For position guidance, it tracks the center of the oncoming lane.

The transition from DropBack to DropBehind indicates that the vehicle has moved behind the lead vehicle. The predicate associated with this transition evaluates to true once the distance between the vehicle's front bumper and the lead vehicle's rear bumper exceeds some threshold.

DROPBEHIND HCSM

This state indicates the vehicle has moved behind the lead vehicle and is ready to merge back from the oncoming lane to the lane where it started the overtake maneuver. For position guidance, the state maneuvers the vehicle laterally from the oncoming lane to the original lane. For velocity guidance, this state uses the Follow HCSM.

The transition from DropBehind to Done indicates that the abort maneuver is complete and the vehicle is back in its original lane. The transition's predicate evaluates to true once the vehicle has reached the center of the original lane.

CONCLUSION

In this paper, we have described the design of an autonomous vehicle's overtake maneuver. This interesting and complicated behavior has been implemented using HCSMs. This exercise demonstrates how their hierarchical and concurrent properties make them a natural framework for implementing complicated behaviors that require the evaluation of several competing goals. HCSMs are a good solution for implementing the realistic behaviors users expect from interactive agents in today’s virtual environments.

REFERENCES

[1] Boer, E.R., & Hoedemaeker, M. (1998). Modeling Driver Behavior with Different Degrees of Automation: A Hierarchical Decision Framework of Interacting Mental Models. In Proceedings of the XVIIth European Annual Conference on Human Decision Making and Manual Control.

[2] Schmidt, K., & Lind, G. Dynamic Traffic Simulation Testing. Traffic Technology International, Dec 1999/Jan2000, 54-57.

[3] Zarrillo, M.L., Radwan, A.E., & Al-Deek, H.M. (1997). Modeling Traffic Operations at Electronic Toll Collection and Traffic Management Systems. The International Journal on Computers and Industrial Engineering (special edition, ed. Y.D. Kim), 33(3-4): 857-860.

[4] Cremer, J., Kearney, J., & Papelis, Y. (1995). HCSM: A Framework for Behavior and Scenario Control in Virtual Environments. ACM Transactions of Modeling and Computer Simulation, 5(3): 252-267.

[5] Papelis, Y., & Ahmad, O.A. Comprehensive Microscopic Autonomous Driver Model for Use in High Fidelity Driving Simulation Environments. Submitted to the Transportation Research Board Conference, Washington, DC. January, 2001.