DSC North America 2003 Proceedings, Dearborn, Michigan, October 8-10, 2003 (ISSN 1546-5071).

Event Driven Dynamic Interactive Experiments (EDDIE)

Jason R. Williams
AAI Engineering Support, Inc.
Turner-Fairbank Highway Research Center
6300 Georgetown Pike
McLean, VA 22101
(202) 493-3392
jason.williams@fhwa.dot.gov

Duoduo Liao
AAI Engineering Support, Inc.
Federal Highway Administration
Turner-Fairbank Highway Research Center
6300 Georgetown Pike, Room T-0119
McLean, VA 22101
(202) 493-3393
duoduo.liao@fhwa.dot.gov

John A. Molino
Science Applications International Corporation
Transportation Research Division
Turner-Fairbank Highway Research Center
6300 Georgetown Pike
McLean, VA 22101
(202) 493-3381
john.a.molino@saic.com

Vaughan W. Inman
Science Applications International Corporation
Transportation Research Division
Turner-Fairbank Highway Research Center
6300 Georgetown Pike, Room
McLean, VA 22101
(202) 493-3380
vaughan.inman@fhwa.dot.gov

Gregory W. Davis
Federal Highway Administration
Office of Safety RD&T
6300 Georgetown Pike, Room T-0119
McLean, VA 22101
(202) 493-3367
gregory.davis@fhwa.dot.gov

John M. Wink
AAI Engineering Support, Inc.
Federal Highway Administration
Turner-Fairbank Highway Research Center
6300 Georgetown Pike, Room T-0119
McLean, VA 22101
(202) 493-3413
john.wink@fhwa.dot.gov

DSC North America 2003 Proceedings, Dearborn, Michigan, October 8-10, 2003 (ISSN 1546-5071).

September 12, 2003

DSC North America 2003 Proceedings, Dearborn, Michigan, October 8-10, 2003 (ISSN 1546-5071).

ABSTRACT

The Highway Driving Simulator (HDS) at the Turner Fairbank Highway Research Center (TFHRC) is used to conduct research on how drivers perceive and react to the roadway infrastructure. The simulator allows researchers to test how drivers will react to the current roadway infrastructure and to new infrastructure innovations. The previous generation of the HDS used linear scripting and hard coded algorithms combined with preloaded geometry to generate and control interactive three-dimensional (3D) scenes. In these earlier simulations the scenario file stored the adjustable parameters and desired 3D models for the scenes. However, these techniques are not well suited to script and define the situations required for certain kinds of current research simulations used at the TFHRC. The earlier scenario files did not control the actual progress of the simulation, but rather defined the environment in which the driver interacts. A variation of the traditional scenario control approach is described in the present paper. The Event Driven Dynamic Interactive Experiment, or EDDIE, method expands on various techniques developed for other high fidelity driving simulators, as well as for certain 3D gaming rendering engines. The EDDIE method uses an event driven state engine, not only to control the states of objects, such as autonomous traffic or traffic signals, but also to control the actual progress of the scenario, including which stimulus is given to the driver and when. Such a mechanism to dynamically control the progress of the experiment is required for research which concentrates on features of the physical roadway infrastructure. For this type of research, data collection events are defined only for situations in which data needs to be collected. Events are also defined to generate specific experimental conditions, such as a red-light running vehicle designed to elicit a specific driver reaction. Defined and derived events form the basis for a scenario. These events are combined with scene objects and content definitions, each with its own states and events. Event driven experimental scenarios replace what would otherwise be a sequential series of scenes. In these event driven experiments, randomization functions, such as Monte Carlo algorithms, are used to order the experimental conditions in an unpredictable presentation order. The use of specific event conditions for triggering data collection reduces data storage to only those data required by the researchers, and facilitates quick-look plotting of important results. The EDDIE scenario control software links the discrete event model with real-time interactive driving through the simulation. The EDDIE software has been used for studies concerning nighttime curve recognition and intersection collision avoidance. These two types of studies serve as examples of how the EDDIE scenario control software uses events to define the scenes, control the experimental sequence, specify data collection opportunities, and provide quick-look results.

INTRODUCTION

The Highway Driving Simulator (HDS) at the Turner Fairbank Highway Research Center (TFHRC) is used to create experiments in which a computational vehicle dynamics model is interactively driven through a virtual reality environment, including traffic, roadway, signals, signs, pedestrians and other environmental characteristics. The basic goals of the HDS are similar to the system requirements of certain early driving simulators (1), which have grown more complex and realistic over the years. Such simulators have been extensively employed to support a variety of training, education and other applications, ranging from traffic safety research to the test and evaluation of vehicle designs. The experiments that the HDS supports vary widely as to the type of driver stimuli required, including motion, visual, auditory and/or force feedback. The scenarios often need to be repeatedly modified; and the scene objects are often dynamic in nature, with many states that must be represented. The scene objects may require that new states be generated to simulate, for example, a nearby traffic control device. These kinds of requirements suggest the use of event driven finite state machines and dynamic scenario control mechanisms as the best way to implement HDS experiments. This approach is a departure from the previous system used to create HDS scenarios, which consisted mainly of a flat file script to define the sequence of the simulation.

Historically scenario controls have had a long evolution and have improved over the years. The Scenario Definition Language (SDL) allows users to define the visual database, interactive traffic and other event sequences, and to collect performance measures with simple text files (2, 3, 4). In the early 1990's, much work on discrete-event dynamic systems was done in systems theory and simulation methodology (5, 6, 7). Most driving simulation systems today, such as the Iowa Driving Simulator (IDS) (8, 9), the BMW simulator (10), and the Traffic Research Center (TRC) driving simulator (11), and others (12), use event driven scenario control models for driver behavior research and other driving applications. The IDS proposed a framework, Hierarchical Concurrent State Machines (HCSM), for behavior and scenario control (9). Such a scenario control system ensures that each state machine is executed on schedule. With this system the road database and scene rendering are updated at the end of each computation cycle.

A variation of this scenario control approach is proposed in the present paper. The Event Driven Dynamic Interactive Experiment, or EDDIE, method expands on various techniques and innovations developed for other driving simulator systems and 3D game rendering engines. EDDIE represents a particular solution, which merges dynamic scenario controls with event driven techniques designed to meet the requirements of behavioral research in driving simulation experiments. The present paper discusses how these techniques were integrated into a single scenario control system to answer some of the unique research requirements at the TFHRC.

BACKGROUND

The Highway Driving Simulator at FHWA-TFHRC

The Highway Driving Simulator (HDS) at the US Federal Highway Administration's Turner-Fairbank Highway Research Center (TFHRC) provides realistic driving simulation capability for use in human centered research. The simulator is depicted in Figure 1. The characteristics of the HDS and the experiments which have been run at TFHRC in the past several years form the basis for the EDDIE scenario system. A brief description of the HDS system and recent HDS experiments will help in understanding the background for the development o f the EDDIE system at the TFHRC.

The current FHWA HDS system consists of a fully-interactive vehicle, a 3-degree of freedom motion base, a high-end graphics workstation, a digital audio system, one projector, and a curved screen. The curved projection screen in front of the car cab provides an 88-degree horizontal field of view of the simulated roadway environment, with a vertical field of view that encompasses the entire windshield. Scene elements are generated by both real-time scene graph modeling and dynamic scene control programming.

Figure 1: The Highway Driving Simulator

Experiments in the HDS

The first experiment conducted in the HDS following a recent major upgrade was a visibility experiment. This experiment was conducted in the spring of 2002. This visibility experiment consisted of a large number of trials in which pavement markings were varied on simulated curves in a night driving scenario. In this experiment, the participant indicated the direction of the curve ahead on the current roadway segment, and the next segment was presented immediately after the response. A flat file (i.e., a text-based data file) described the sequence of roads and the associated parameters for each roadway segment. The roadway segments were presented sequentially in a randomized order. The road scene was composed of a single Multigen™ Creator™ 3D model file in which all the variant road geometries, pavement markings and raised pavement marker conditions were modeled. There were over 100 different experimental conditions that were manually modeled. This method, while straight forward, required a large amount of modeling time, and was prone to human error, because parameters had to be linked to an index offset position in the model file.

In a similar experiment that followed (as illustrated in Figure 2), an event-based dynamic scenario development method was introduced, and a scenario sequence file was used to construct the scene elements. This included loading curves based on angles of deflection and degree of curvature. Then, as the experiment ran, the correct roadway geometry, as defined in the scenario sequence file, was generated on the fly. Similarly, modeled retro-reflectivity of pavement markings and raised pavement markers were generated during scenario presentation based on an order defined by the researcher. This method greatly reduced the element of human error from scene generation and required much less development time.

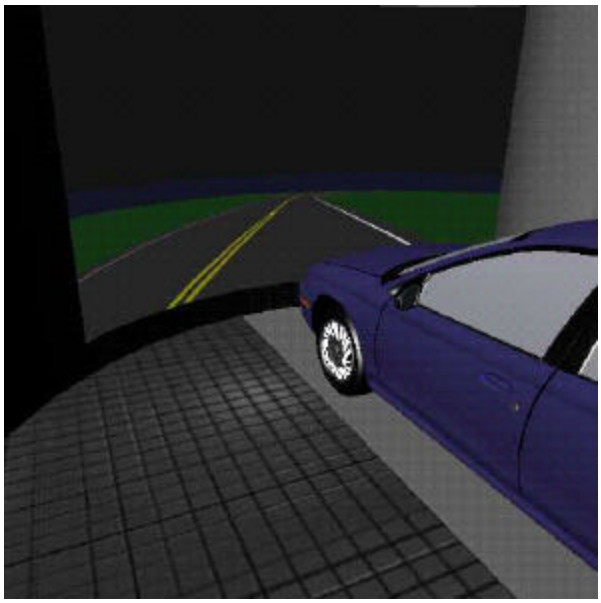


Figure 2: Curve Performance Study

A future experiment will evaluate alternative intersection collision avoidance countermeasures, as illustrated in Figure 3. In this experiment, participants will drive through multiple intersections in which traffic signal phase will be varied and various collision avoidance warnings will be given. The intersection model is based on an actual rural intersection in Northern Virginia. The CAD drawings for the intersection were obtained from the Virginia Department of Transportation. Digital photographs were taken at the Northern Virginia location to provide texture elements. Variants of the intersection were created to form a series of intersections. Each variant was defined by the use of a library of textured 3D elements. These elements were created based on objects at the real-world Northern Virginia site, and were modeled in the Creator™ 3D real-time software tool. In addition there was a need to create a scenario control system that could quickly change variables and parameters, such as traffic signal phases, visibility-scaling controls, threat vehicle behavior, and warning controls, based on vehicle positions. Dynamic data collection events were also needed. These were based on parameters such as traffic signal phase and where the driver was relative to the signal.

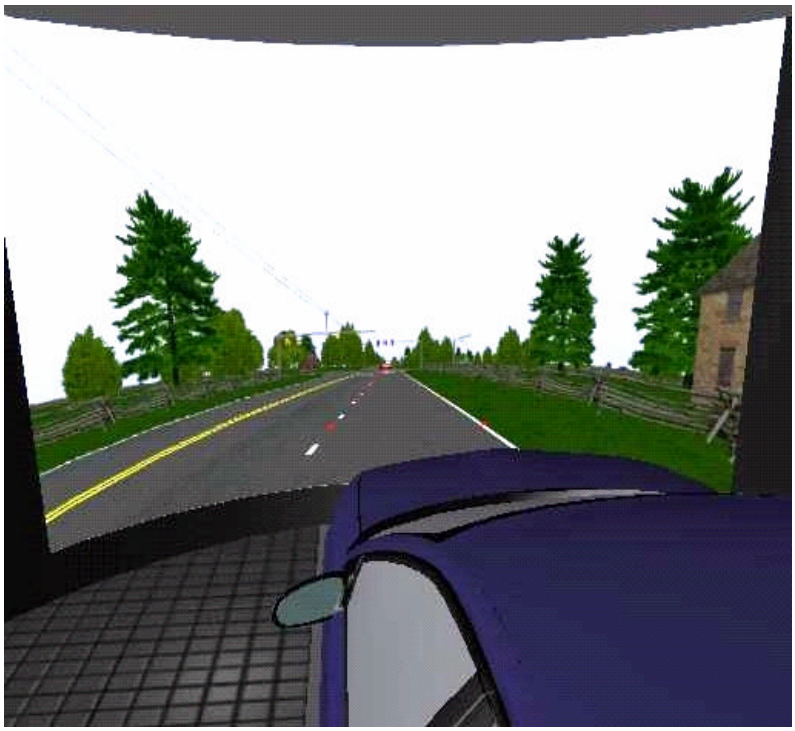


Figure 3: Intersection Collision Avoidance Study

Research Approach Requirements

From a high-level systems engineering perspective, whether a simulator is used for training or for research, the problems are similar. To achieve defined goals and objectives a system must fulfill certain requirements. Some of the key requirements for the current research are to:

- Adapt to frequently changing research needs;
- Implement experimental manipulations free of potentially confounding stimuli;
- Provide precise measurements of driver responses to discrete, well defined stimuli; and
- Present stimuli in a way that provides reliable generalization to the corresponding real-world situation.

Research requirements are continually changing. If a research study is successful, the need to duplicate the same conditions in the simulator may be unnecessary. Thus scenario generation consumes a much larger proportion of the simulator's life cycle than in the case of a training simulator, where the students, rather than the scenarios, are more likely to change.

When comparing two experimental treatments, as in testing alternative ways of warning drivers of an imminent collision with a red-light runner, it is important that only the treatments vary between trials. For instance, a current experiment evaluates whether flashing red beacons or flashing Light Emitting Diodes (LEDs) embedded in the roadway are more effective in alerting motorists to stop. If the LEDs only flash at an intersection next to a barn, and the beacons at an intersection next to a church, then a possible confounding effect may be introduced. Scenarios that are difficult to modify often result in just such confounds.

The research on warning the potential victims of red-light runners, investigates which warnings, if any, will induce a large proportion of motorists to stop quickly in a situation where they would normally be reluctant to stop at all. The warning must be effective without previous training, and it must be effective the first time it is experienced. In this evaluation the warning comes when the motorist is very close to the simulated intersection. The warning must occur at a precise moment. Accelerator and braking responses to the warning must be measured with millisecond accuracy. Variations in seemingly trivial things, such as the refresh rate of the Cathode Ray Tube, may affect the reaction time results or mask real effects.

Finally, there must be correspondence between driver response in the simulated systems and the real world. This requirement to generalize to the real world is typically the most difficult obstacle to overcome in simulation. Warnings need to be as conspicuous in the simulation as they would be in the real world, and they need to be perceivable at the same distances and in the same amount of time. Although it is occasionally possible to mathematically scale stimuli or responses to equate for differences between virtual and real worlds, the scale factors are seldom known in advance. The range in which such factors are applicable is rarely determined. Thus, research simulator fidelity is a continuing challenge.

Trial Based Research Requirements

The research at the TFHRC often requires performing multiple trials of data collection. Scenarios created with the EDDIE software must be able to run multiple trials of very similar scenes repeatedly with small changes to the experimental variables. This requires simulation procedures that can collect a large amount of data in a short period of time. Therefore, instead of long naturalistic driving scenarios, a more efficient trial-based methodology was developed. This methodology is implemented in two ways. The first type of experiment measures the visual recognition distance to important roadway features dynamically viewed from afar. The second type of experiment measures the performance of drivers as they actually navigate through the roadway feature under investigation.

In the first type of experiment visual detection or recognition distances are measured while the research participant is engaged in a driving task. The research participant drives simulated straight roadway segments until an important roadway feature can be detected or recognized. This feature may be an upcoming curve, interchange or intersection in the roadway, or any other safety feature requiring advanced warning to the driver. A trial-by-trial interactive driving task is employed. Once the target roadway feature has been detected or recognized, a new trial begins. The driver may never actually arrive at the simulated target roadway feature. Therefore, mainly straight roadway segments

are modeled and no motion base is required. Trials can typically be about 15 seconds long. The dependent variables are usually feature detection/recognition distance, also called preview distance, and vehicle driving speed. This methodology results in far fewer stimulus presentation trials than are needed with other psychophysical determinations of preview distances. Also, the research participant's task is more realistic and the allocation of attention more accurate than with static picture viewing techniques using photographs of roadway features.

In the second type of experiment the performance of drivers is measured as they actually navigate through the roadway feature under investigation. Like before, the research participant drives simulated roadway scenarios that contain important roadway features. However, this method employs a quasi-continuous driving scenario. The research participant not only detects or recognizes the simulated target roadway feature from a distance, but actually drives through it. This method requires the research participant to execute complex driving maneuvers to navigate the roadway feature of interest (curve, interchange, intersection, etc.). Meanwhile, driving performance measures are recorded (vehicle speed, acceleration, path, lane position, etc.). This method provides the most realistic driving task for a simulated environment. It requires interactive visual, auditory and motion cues, since the participant is negotiating complex simulated driving situations. It also takes more time to execute a single trial, but still not as much time as a naturalistic driving scenario. Trials can typically be about 30 seconds to one minute long. Although this method appears to the driver as rather continuous, the roadway conditions do change on each trial.

Both of these types of trial-based experimentation require the minimally perceptible changing of simulator scenes at precise times. These times are dependent upon events programmed into the scenario itself, or generated by the research participant serving as the simulator driver. Both types of experimentation also require that the roadway geometries of successive scenes be precisely aligned, so that trial transitions appear as seamless as possible to the driver. The EDDIE scenario control software is designed to fulfill all of the above requirements.

Event Based Research Requirements

In the research conducted at the TFHRC, the simulation has to quickly adapt to highly dynamic situations in order to accommodate specific data collection requirements. That is, behavioral measurements may only be relevant when they are connected with specific events in the simulation run. For example, occurrences of brake application may only be relevant when a red traffic signal light is active and the vehicle is within a specific range of speed and distance from the intersection. As mentioned previously, one experiment in the HDS laboratory is investigating the effects of administering infrastructure-based warnings to drivers of a potential red light runner. For this experiment, the researcher requires the research participants of the experiment to drive through many intersections to create an expectation. The last intersection, i.e. the test intersection, is the location in the simulation run where the researcher is most interested in driver behavior. Specifically, the researcher is interested in collecting reaction time data when the driver is presented with an unexpected warning device placed near the intersection. Because it is a surprise event, data need only be collected once per participant. Thus, a trial based scenario where research participants are repeatedly exposed to various stimuli was not feasible. Under these circumstances, an event-based scenario proves to be a superior approach for conducting this type of experiment. An event trigger is placed 225 ft in front of the last "test" intersection. As the participant drives past this event trigger while maintaining a speed of 40-50 mph, the warning devices in the intersection are activated, and data collection begins. The event based scenario development allows for more efficient post-processing of the data.

Scenario Definition and Scene Content

The following definitions of simulation modeling elements and how they are being used will help to explain the components of the EDDIE design. The scenario is defined by scene content and trial parameter requirements. Evaluation of the scenario to be run can be quite helpful in understanding what is needed and what parameters must change dynamically. Experiments that vary stimuli on each trial need to be driven based on the parameters that control the stimuli for those trials. These parameters may require entirely different scene content such as a different curved road section or a different intersection. The experiment itself can be designed to be discontinuous from a scene content perspective, i.e. jumping from one curved segment approach to a different curved segment approach

without a connecting roadway, or without even connecting the curved segments. However, scenarios that have continuous scene content are probably more realistic and less disconcerting to the research participants. The description of an experiment is defined by a sequence of scenes to be rendered, i.e. driving down a straight road for a set distance, then taking a curve with a specified curvature, and then stopping at a 4-way intersection with a red light, etc. Each of these separate scene descriptions can be used as the independent components of the scene. Parameters can be added to control additional scene states such as traffic light signal states, pavement markings, road signs, and other traffic behavior. The smallest level of scene detail needs to be based on what is being controlled and tested in the experiment. These detailed components can then be reused with changes to scene element settings (parameters) as needed, and will be easy to employ again in future experiments. Variants of scene elements, which have multiple appearances, can be represented by using the 3D modeling mechanism of a switch node to represent differing states. In this way differing appearances can be parameterized, and then described in the scenario file by the mask setting value used to render it with the associated appearance. Differing appearances can be dynamically set when a trial calls for a different ensemble of pavement markings on the road segment. This method works for the majority of dynamic states in most scene content, but more intricate mechanisms may be needed for more complicated dynamic behavior, such as pedestrian or vehicle autonomous behaviors.

When the states are changed, discrete event simulation modeling (13) defines the actual changing of state as an event. In fact the entire experiment scenario may be considered as a series of events that occur in the simulation. Some of these events are triggered based on the driver's actions or location, and other events are triggered automatically based on the scenario definition.

Dynamic Scene Elements

Most research projects conducted in the HDS require dynamic scene elements with characteristics that need to vary in a controlled way from trial to trial and driver to driver. These elements may be related to actual scene content, such as traffic signal lights, or to scenario controls that are not visible elements in the scenario. The researchers require control of aspects of the simulation at a finite level on both the scene characteristics and on the dynamic scenario controls. The scenario control application must be able to change these aspects as needed. The application must control parameters and the way these parameters interact with the driver. A simple example of this might be to simulate a dynamic message sign that displays the speed to the passing driver. The specific parameters may not be known ahead of time, requiring the scenario system to be flexible. These requirements, and the need to reuse previously built scenario elements for quickly creating scenarios, suggest that the system should use object-oriented methods. Object-oriented methodologies are used to create systems which are more reusable, flexible, extendable, and naturally modeled (14).

A SCENARIO MANAGER DESIGN

Besides an object-oriented approach to scenario management design, the next most important determinants of how to define and control scenarios are the common elements among various experiments. Understanding the common elements is achieved by analyzing the experiments as components based on: 1) physical scene elements (visual, motion, and auditory), 2) experimental parameters that are being tested and/or presented, and 3) the independent discrete events that are used either as scenario control elements and/or data collection opportunities.

Figure 4 represents the results of the current design effort. This Figure depicts a software engineering object diagram, showing the relationships among the system components of the scenario manager.

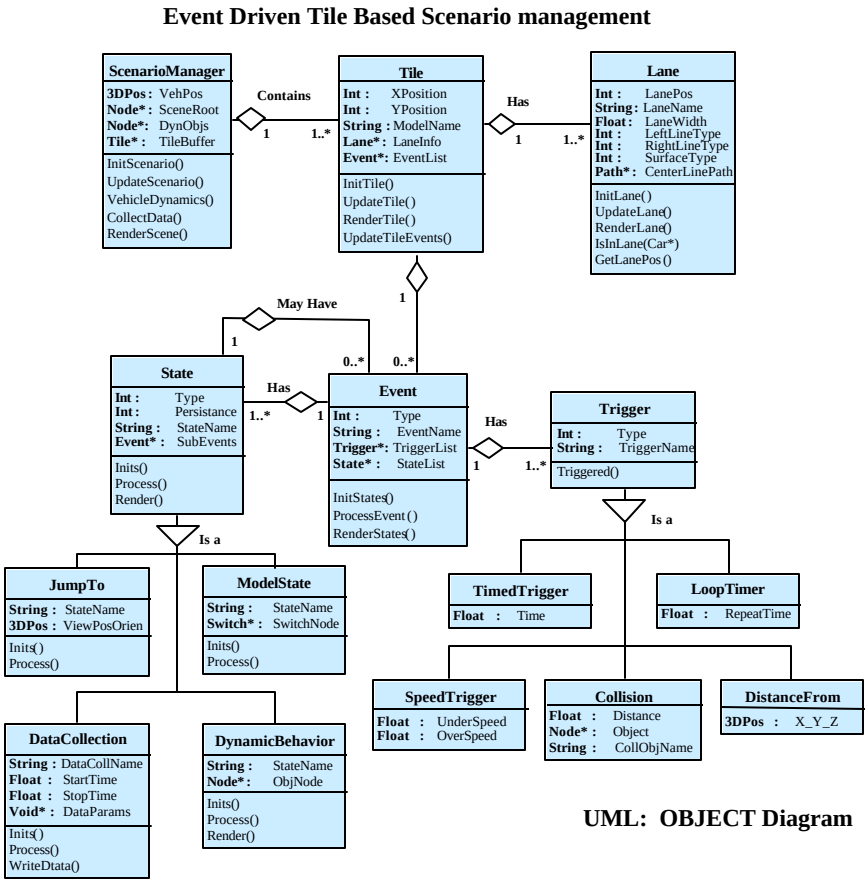


Figure 4: Event Driven Design

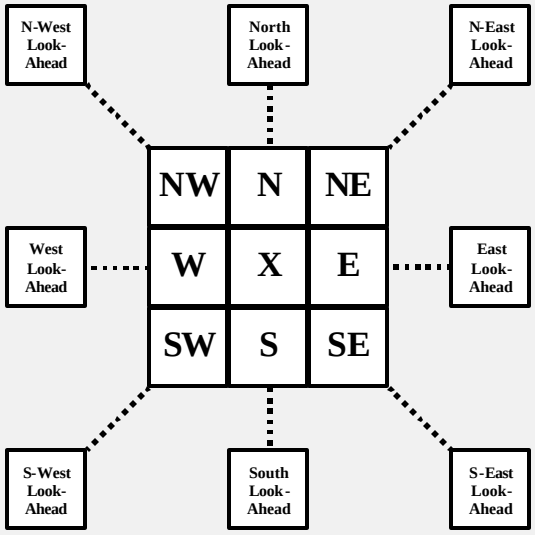
Scenario Manager

The scenario manager is the main component of the simulation system. The scenario manager updates the scene based on what is needed as defined in the scenario definition files. When the scenario manager is initialized, it loads and initializes all scene elements (tiles and external model references), event lists, and lane definitions. Each of these objects is described later in more detail. These objects contain all the definitions on how to run and control the scenario. The scenario manager processes all scenario events for each rendered frame. The graphics rendering, I/O control systems, vehicle dynamics model, and data collection subsystems are not shown here, but do interact with the scenario manager subsystem. The main data structure that defines the scenario is the tile list. The scenario manager receives information on tile contents, the initial position for the scenario, and also the total size of tiles. The tile look-ahead size is also predefined in the scenario file.

The Tile Mechanism

The tile system has been used in simulators and game systems since their introduction. The tiles are used to store geographic chunks of the scene by dividing the virtual world (play area) into equal sized grids. Each grid is considered a tile. Dividing the scene into tiles allows the software to buffer more manageable pieces of the geometry needed for the scene and, in doing so, allows a much larger overall area for the simulation. Tiling also facilitates subsequent scenario generation through reuse of tiles previously generated.

Each tile is linked to an OpenFlight™ 3D model file. This model file contains all scene appearance elements for the particular tile, such as trees, buildings, signage and pavement markings. Performer™/OpenFlight™ switch nodes (and other advanced real-time modeling techniques) are used to select variant states of these elements' appearances. In this way a single OpenFlight™ model can be created to represent many tiles that use similar geometric forms, but require minor differences such as lane markings. The only constraints in using tiles are that roadway paths on the tile edges should always match up, and that tile sizes should be fixed. Both of these constraints limit the use of tiles. The tile definitions should be thought of as controlling the experiment from a positional point of view. When an experiment is run that consists of discrete trials on curves, the tiles contain the tangent and the curve on a given tile. The driver will be presented with the next trial by relocating the driver on the next roadway segment. For continuous driving experiments, such as the intersection collision avoidance system experiment, the set of adjacent tiles defines the driving route that the participant follows. Variant paths can be created based on the experiment's needs and parameters.



X = center Tile where the Simulated Car is being driven.
Look Ahead determines the total buffer size of Tiles to render.

Figure 5: The Tile Buffer

Roadway / Lane definitions

Each tile contains one or more centerline paths or lane definitions (pavement markings, traffic control device positions, super-elevation angles). Lane definitions can be used to measure driver lane keeping for the research participant, or path definition for autonomous traffic. Parameters for pavement markings and other lane-based conditions, such as curvature and deflection angles or road surface characteristics, are also contained in the lane definitions.

Events and Triggers

EDDIE employs features borrowed from discrete event techniques and definitions used in non-real-time simulation, such as factory floor simulation or network simulation. Event based programming has been introduced into simulation and entertainment applications, especially in games that contain artificial intelligence algorithms. In discrete event simulation, an event is anything that happens which changes the state of an object. These state parameters normally change with regard to the event time. When the simulation is run, each event is processed sequentially as read from the time queue. Processing an event can cause new events to occur, but only in current time or future time, never in the past. Such an operation places new events in the event queue.

A trigger is a controller object that initiates the occurrence of a specific event when certain conditions are met. For example, a data collection event needs to be triggered when the vehicle passes within a given distance of a crosswalk in the roadway. Events can have one or more triggers. Triggers are generalized objects that can be subcategorized by time, position, distance, lane keeping, looped timers, speed checks, sounds, or collision events. Furthermore, through object-oriented definitions, the basic object type can be extended to new kinds of triggers as needed.

States

Events are associated with scene elements that have a change of state. For instance, a collision event could be associated with several states, including crashing sounds, the appearance of a car being smashed, and a zero velocity condition on the vehicle dynamics model. Through effective state modeling, it is possible to define just about any scenario an experiment needs. States can be associated with multiple objects in the scene, or with the appearance of a given tile. The state could be an instantaneous position change of the vehicle, as needed for jumping to a new roadway segment. States may be continuous and persistent, or instantaneous and non-persistent. An event can be used to turn on a data collection state, for instance when the driver passes through a yellow light. States can also be applied to control the dynamic behavior of the other vehicles or pedestrians represented in the simulation. Some algorithms for these states/event models can represent complex behavior, and can include complex roadway devices such as loop detectors or other more intricate sensing devices.

CONCLUSION

The EDDIE scenario control software as presented here represents the conceptual design for the FHWA HDS system. Various parts of the scenario control software have been implemented but more work needs to be done to implement the full system. The HDS staff is continually learning and leveraging techniques from the simulation, 3D gaming, and real-time 3D graphics industries for incorporation into the design and implementation of the HDS system. Areas for further work include a GUI for creating and modifying the scenario, a more robust finite state machine, and a mechanism to communicate current and changed states with networking packets. A GUI will provide non-programmers with a simple interface to create and modify scenarios, allowing better leveraging of resources to construct the scenarios for experiments. Optimization and borrowing from finite state machine definitions used in the gaming industry will enable the use of more complex state tables. A future upgrade of the simulator system will require the use of multiple PC-based rendering computers. This will require the scene states which affect the appearance of rendered elements to be communicated across all rendering machines. This last enhancement will also enable the use of multiple simulators to operate together (multi-play).

ACKNOWLEDGEMENTS

The work reported herein was conducted for the FHWA under two separate contracts. One contract was with AAI / Engineering Support, Inc. The other was with SAIC. The authors wish to express special appreciation for guidance and direction from M. Joseph Moyer, Thomas M. Granda and Gabriel Rousseau of the FHWA Human Centered Systems Team, and from Barry Wallick of AAI/ESI and Ron Huffman and Stephen Greene of SAIC. The authors are grateful to all who helped to make this endeavor a success.

REFERENCES

1. Rod Deyo, John A. Briggs, Pete Doenges. Getting Graphics in Gear: Graphics and Dynamics in Driving Simulation. ACM Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'88). June 1988.

2. A.C. Stein, R.W. Allen, et al. Applications of Low Cost Driving Simulation, Proceedings of Driving Simulation Conference (DSC'95), Sophia Antipolis, France, 1995.

3. R. Wade Allen, Theodore Rosenthal, Bimal Aponso. Low Cost, PC-Based Techniques for Driving Implementation. Proceedings of Driving Simulation Conference (DSC'99), Paris, France, 1999.

4. Peter van Wolffelaar, Salvador Bayarri, Inmaculada Coma. Script-based Definition of Complex Driving Simulator Scenarios. Proceedings of Driving Simulation Conference (DSC'99), Paris, France, 1999.

5. Y. Brave and M. Heymann. Control of Discrete Event Systems Modeled as Hierarchical State Machines. IEEE Trans. Autom Control 38, 12 (Dec.) 1803-1819. 1993.

6. M. Heyman. Concurrency and Discrete Event Control. In Discrete Event Dynamic Systems: Analyzing Complexity and Performance in the Modern World, IEEE, New York, 1992, pp.65-75.

7. P. Ramadge and W. Wonham. The Control of Discrete Event Systems. In Discrete Event Dynamic Systems: Analyzing Complexity and Performance in the Modern World, IEEE, New York, 1992, pp.48-64.

8. J. Cremer, et. al., "The Software Architecture for Scenario Control in the Iowa Driving Simulator," Fourth Conference on Computer Generated Forces and Behavioral Representation,
May 4-6, 1994, Orlando, FL, pp. 373-381.

9. James Cremer, Joseph Kearney, Yiannis Papelis. HCSM: a Framework for Behavior and Scenario Control in Virtual Environments. ACM Transactions on Modeling and Computer Simulation (TOMACS), Volume 5, Issue 3. July 1995.

10. M. H. Strobl, J. H. Bernasch, J. P. Lowenau, “Generation of Complex Traffic Scenarios in the BMW Driving Simulator,” Proceedings of the Driving Simulation Conference, 2000, Paris, France, September 6-8, 2000, pp.245-256.

11. P. C. Van Wolffelaar, et. al., “Traffic Modeling and Driving Simulation – An Integrated Approach,” Proceedings of the Driving Simulation Conference (DSC ’95), 1995, Sophia Antipolis, France,
September 12-13, 1995, pp. 236-244.

12. Y. Papelis, O. Ahmad, “A Comprehensive Microscopic Autonomous Driver Model for Use in High-Fidelity Driving Simulation Environments,” Proceedings of the Annual Transportation
Research Board Meeting, Washington, DC: TRB, 2001.

13. F. Oliver Gathmann. Python as a Discrete Event Simulation Environment. Sept. 10, 1998. http://www.foretec.com/python/workshops/1998-11/proceedings/papers/gathmann/gathmann.html. Accessed July 18, 2003.

14. Ricardo Devis. The Object-Oriented Page. June 20, 1997. http://www.well.com/user/ritchie/oo.html. Accessed July 16, 2003.