ENABLING TECHNOLOGIES FOR EMBEDDED SIMULATION & EMBEDDED TRAINING

Hubert A. Bahr
HQ STRICOM AMSTI-ES
12249 Science Drive Suite 1403A
Orlando, FL 32826
407-384-3874
Hubert_Bahr@stricom.army.mil

Claude W. Abate
Sherikon, Inc
12249 Science Drive Suite 140
Orlando, FL 32826
407-384-5397
CAbate@Sherikon.com

Keywords:

Embedded Simulation, Embedded Training, Autonomous Trainers

ABSTRACT

The army has placed a renewed emphasis on an embedded training capability as a result of lessons learned from the Advanced Warfighting Experiment (AWE) 97-06 on the potentials of digitization. Through the Inter-Vehicle Embedded Simulation Technology (INVEST) Science and Technology Objective (STO), the Simulation Training and Instrumentation Command (STRICOM) will develop the technology that will lay the foundation for incorporating embedded simulation into future and legacy combat vehicles. This paper presents current status and future evolution of the enabling technologies needed to fully embed these technologies into a combat vehicle. These ES systems will support both training and operational (go-to-war) enhancements for the Army XXI and Army After Next inventory of combat vehicles. The key enabling technologies for an autonomous vehicle capability include: low cost image generation, live-virtual object and terrain integration, virtual target injection into sensor displays, synchronized semi-automated player models, simulation filtering tool, intelligent tutoring system, time-based and UWB communication, and automated vehicle model development and optimization.

ENABLING TECHNOLOGIES FOR EMBEDDED SIMULATION & EMBEDDED TRAINING

By Hubert Bahr and Claude Abate, STRICOM

Introduction

To fight and win on the modern battlefield two things are required; weapons systems that out perform the opponent's weapon system, and crews that are better trained to use the weapon system effectively. A cost effective means of improving weapon system performance is Embedded Simulation (ES), which includes Embedded Training (ET) or the capability to train and maintain crew proficiency on the same equipment they will go to war on, and the Embedded Operations (EO) functions of situational awareness (SA), mission rehearsal (MR), command coordination (CC), critical decision making (CDM) and course of action analysis (COAA).

The competition for better weapons is one component of the challenge and the other involves training the crews to be more proficient than the opponent. But there are training costs associated with more complex weapons systems. To date, stand alone trainers have been employed at the school house and in the units. The power projection army of the future will have to spend more time maintaining task proficiency while stand-alone trainers cannot meet the deploying force requirements and are too costly to operate and maintain. One option that will overcome this deficiency is a training system that is integrated into the vehicle. However, any sub-system that is integrated into the vehicle becomes a luxury unless it provides improved combat effectiveness. For this reason, the proposed approach of embedded simulation (ES) technology addresses a system that provides both training support and go-to-war capabilities. An expanded discussion of ES training uses can be found in reference 1.

In this paper we are going to walk through the various levels of training applications (gunnery) from a single vehicle to multiple vehicles to multiple vehicles participating in live fire and force-on-force exercise similar to training conducted at the Combat Training Centers (CTCs) and finally go-to-war examples.

Background

While stand alone trainers such as COFT/AGTS, SIMNET/CCTT and M-1 Driver Trainers have served the Army of Excellence well, technological advances and miniaturization now present the ability and affordability for embedding crew and collective training systems into the vehicle. We will refer to these ground combat systems with an embedded training capability as "autonomous" trainers.

The goal of the Inter-Vehicle Embedded Simulation Technology- Science and Technology Objective (INVEST-STO) program is to develop and demonstrate the technology that will lay the foundation for incorporating ES and ET into future as well as legacy vehicles.

The enabling technologies and components used to run an Embedded Simulation System (ESS) are basically the same for the stand-alone trainer with the exception that they will be smaller, faster, more powerful and less expensive. Common components include image generators, simulation computer, Semi-automated Forces (SAF), data logger, terrain database, communications and instructor operator. Stand alone Image generators of today are located in large racks and wired to monitors located at the various crew stations. In the future they will be no larger than a card and the images projected directly into vehicle sights or sensors. The large rack mounted computers will be replaced by a very small and powerful lap top size computer that is accessable to the crew and loaded with software (SW) containing current ModSAF and world terrain database models. An application hardware (HW) data logger will be linked into the simulation computer to record crew actions and support AARs. The only common component that is non-applicable to an ESS is a dedicated instructor operator because that task belongs to the vehicle commander or senior cadre personnel.

When units deploy to a combat zone in response to a rapid deployment mission in the next century, the benefits of autonomous

trainers become readily apparent. In addition, the dual-use design of the ESS can be used to enhance operational effectiveness.

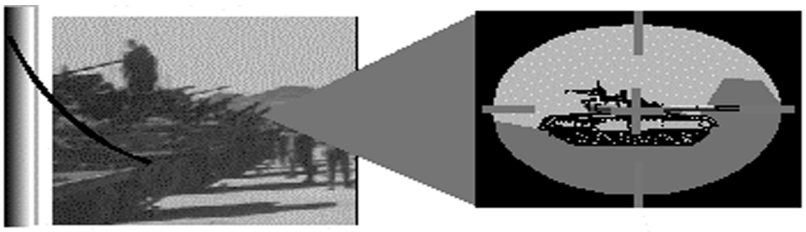


Figure 1A
Assembly Area or Motor Pool Training

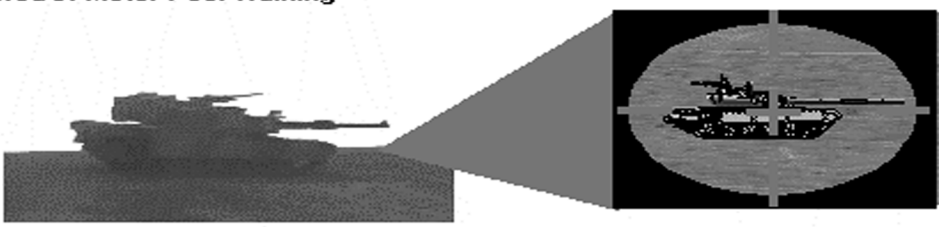


Figure 1B
Range Dry Fire Training

Figure 1C
Range Live Fire Training

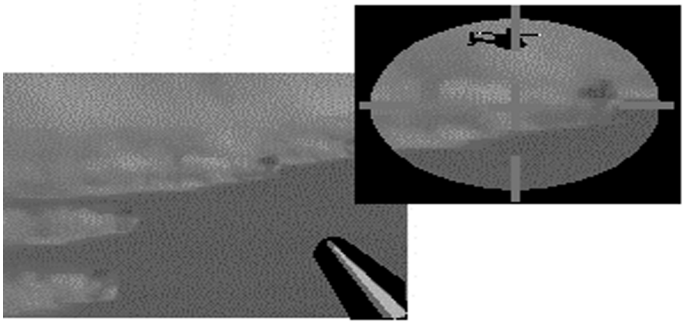


Figure 1D
Force on Force Training

Figure 1E
Combat Operations

Figure 1

ESS Training Applications

Let's assume that in any future combat system the crew will have to be trained to maneuver and engage targets using a highly sophisticated suite of fire control sensors and devices. The design of the ESS will be such that the crew can hone its maneuver and gunnery skills by projecting selected training exercises into vehicle optics or sensor systems. Due to advances in technology and miniaturization, the graphics card and open scene visual processing will be capable of displaying a terrain database (world models support SW), and the SAF (CGF support SW) will display the target array on to the database. These SAF entities will be fully functional (move, shoot and maneuver) and replicate enemy capabilities. The on-board data logger will record all engagements for follow-on After Action Review (AAR) by the unit cadre or vehicle commander. This training can be conducted in the motor pool, assembly area or enroute to a combat theater. Training can be tailored to meet individual or crew (collective training) needs in terms of tactical conditions (offense/defense), force ratios, degree of difficulty in terms of probably of hit & kill, and environmental, terrain and light conditions. This provides a virtual 360-degree battlefield with ground and air targets.

Training Example

Training will follow the normal crawl, walk, run strategy starting with a stationary single crew exercise and progress to multiple moving vehicles in a combined arms live fire exercises. However, autonomous trainers require a further segregation of exercises in terms of simulation mode, i.e. (1) live veh firing virtual rounds vs. virtual target on virtual terrain; (2) live vehicle firing virtual rounds vs. virtual target on live terrain and (3) live vehicle firing live rounds vs. virtual target on live terrain. The technology / engineering challenges associated with each mode are listed below.

1. Live vehicle firing virtual rounds vs. virtual target on virtual terrain requires:

    a. geometric pairing vice laser pairing
    b. aim point determination
    c. realistic fire on target effects
    d. scenario generation

(a) Geometric pairing will be required because a laser pairing system will not work between live and virtual targets (no vehicle present to provide a laser return). In virtual on virtual simulation shooter-target pairing is practically inherent because the locations and orientations of the vehicles and weapons are known almost perfectly in the simulation world. Engagements are simulated by computing the ballistic flyout of simulated rounds and determining where they impact on target or the terrain. The geometric pairing solution takes place at the time of ranging to the target (in the shooter's sight picture). The on board simulation computer calculates the distance to target and stimulates the vehicle to enter the appropriate range return in the gunners sight.

(b) Aim point determination will be calculated by capturing, at the instant of firing, the crosshair location with respect to the target. In a virtual on virtual engagement, the locations and orientations of vehicle are known essentially perfectly (they are synthesized by the simulation) and their orientation relative to each other are easily derived from their world coordinates. The simulation computer knows the relative position of crosshair to the target and stimulates the appropriate burst on target effect. Location of impact is also needed to determine target and casualty effects.

(c) Realistic fire on target effects models are stored as part of the terrain database and will be generated by the IG at the time of round impact on the target. Obscuration, gun recoil and visual tracers will also be stimulated in the sights of the firing vehicle at the time of firing.

(d) Scenario generation would be accomplished at the battalion level and in accordance with published gunnery tactics, techniques and procedures. The scenarios developed in this example, would be a series of crew gunnery exercises or firing tables designed to train or sustain crew proficiency. All targets would be virtual and arrayed to match current enemy fire and maneuver

doctrine. Firing scenarios can either reside on the vehicle simulation computer HW, on a CD-ROM or ported down to the using unit or plugged into the removable storage application HW.

2. Live veh firing virtual rounds vs. virtual target on live terrain requires:

   a. terrain fidelity and terrain correlation
   b. injection of virtual target into a live scene

(a) Terrain fidelity and terrain correlation supports a clear image of the virtual target that is spatially correlated to the live terrain. For example, the virtual target must realistically move over the live terrain and not give the appearance of floating above or sinking into the terrain.

(b) Injection of virtual target into the live scene or augmented reality involves the process of generating virtual images that appear to fit seamlessly into the real-world environment. A critical requirement is image clipping or removal of those virtual images that should be partly or fully obscured by intervening real-world objects.

3. Live veh firing live rounds vs. virtual targets on live terrain requires:

a. GPS location of firer
b. Vehicle/ hull attitude
c. Gun/turret orientation (AZ & Elev.)
d. Safety overwatch (observer console)
e. Aim point determination (GPS Interferometry)
f. Geo-pairing

(a) All vehicles will be equipped with a global positioning system (GPS). This system accurately identifies the location of the firer in terms of its X & Y coordinates and every other friendly or enemy vehicle in the exercise can be tracked and geo-paired for gunnery purposes.
(b) The hull attitude of the virtual target is important to determine the strike of round and to calculate vehicle damage. Orientation of a virtual vehicle influences its vulnerability and ability to identify and engage the live vehicle. The simulation computer on the live platform is generating this information.
(c) Gun orientation further defines the virtual vehicle's ability to identify and engage the live vehicle. The simulation computer on the live platform generates this information.
(d) If this technology is used to replace wooded targetry on live fire ranges at home station or at the CTCs, then it is essential to have tower or safety officer overwatch in order to see the live engagement of a virtual target. Safety over watch can only be accomplished by providing a safety-overwatch console with the virtual target array similar to those on the firing vehicle. A simulation computer and image generator (IG) capability must be available to safety personnel or sent by telemetry from the firing vehicle to overwatch element sensors.
(e) When crossing the boundary between live and virtual (or engaging real targets that cannot be seen) the orientation of the real shooter vehicle with respect to the world becomes important. This situation requires solving the problem of measuring accurately not only the position of the shooter and target vehicles, but also the pointing angle in world coordinates of the shooter's gun.
(f) In the real world of live-instrumented vehicles on training ranges, it is not possible to determine locations and orientations very well. Geometric pairing from shooter to target is determined using geometry, namely the locations of the vehicles, the pointing angle of the shooter's gun and the line from shooter outward toward the target.

1. ESS in support of multi-echelon combined arms (collective) training conducted at our Combat Training Centers requires an additional set of enabling technologies. (The fact that multiple players are participating means communication links become pacing items for successful execution). These technologies include:

a. Digitized terrain database

b. Optimal live/virtual registration (Geometric pairing combined with GPS, vehicle attitude and gun orientation)
c. Synchronized SAF
d. Increased communications bandwidth / reduced commo / distributed processing
e. Automated vehicle / smart models
f. Communications / sensor surrogates
g. Intelligent tutoring system
h. Automated battlefield information filtering tool
i. Scenario builder / modification tool

(a) High resolution digitized terrain databases are essential for any live-virtual exercise. Resolution must be to Digital Terrain Elevatoin Data (DTED) level 4 with horizontal resolution at 5 meters and vertical resolution at 1.5 meters or less. All databases should be standardized & interoperable or compliant with Synthetic Environment Data Representation & Interchange Specification (SEDRIS) conversion mechanisms.

(b) Implementation of geo-pairing for direct fire and non line-of-sight engagement simulation is base upon accurate GPS position location measurements and accurate GPS-based turret angle measurements (shooter-target pairing and accurate visual representation of live vehicles in the virtual world).

(c) Synchronized SAF or the Collective Observation of a Common Entity (SAF) is designed to synchronize the SAFs generated on each player platform. For example, in a platoon exercise all tanks will see exactly the same view of the Opposing Force (OPFOR) as they move or as they are attrited by platoon direct fire. The advantage of synchronizing is the reduction of update communications traffic between friendly players as actions take place affecting the status of the SAF entities. The technology involves the modeling of the SAF entity at a high level (in terms of behaviors) so that only infrequent updates to the model are required.

(d) ESS has more stringent commo requirements than any stand-alone system. These requirements include weight, volume, range, power, and bandwidth management. With multiple players in the simulation exercise they must be constantly reporting current state (a few updates per second at a minimum) and interaction information to ensure proper representation. The use of ultra wideband (UWB) technology shows promise with handling the high data bandwidth load. It uses less power for given range, has urban environment capability and a low probability of detection & interception. This UWB technology also has the potential for the longer term “go to war” tactical internet communications requirements.

(e) By accurately modeling the behavior of a human player, each live or virtual entity can use this behavior model to predict the state of other live entities on the battlefield; and thus reduce the commo bandwidth required to update operational and status information exchange between all entities. The optimization of model information can be using on-board computational resources (simulation computer and vehicle modeling SW).

(f) Communication and real-time sensor surrogates will employ UWB technology. UWB has the capability to be used as precision radar with the inherent benefits as a communication system. Use of this capability in an imaging array can provide an electronic video camera and provide a more accurate live/virtual vehicle registration; as well as providing terrain database updates that were not present when the database was produced and thus avoiding costly high resolution database generation.

(g) A SW application connected into the simulation computer system will be an embedded intelligent training system. The SW will duplicate as closely as possible the trainee undergoing on-the-job training in a crew position task environment without benefit of a human instructor. Students can train in an interactive environment towards a particular goal or task that will include challenging training scenarios, monitoring & evaluation of the trainee actions, meaningful feedback comments to errors and response to trainee requests for information.

(h) A SW application connected to the simulation computer system will be designed to process tactical information and/or use intelligent agents to filter out

extraneous information not readily needed by the commander. This system will automate the collection & dissemination of critical information automatically allowing rapid decision-making. This system will prevent information overload by eliminating non-essential information, reducing communications bandwidth, and uncluttering the commander's display.

(i) The TRADOC community will provide a standard library of ARTEP scenarios and the units will have the capability to develop their own scenarios or modify existing ones to meet METL requirements. This technology will be located at battalion staff level and interfaced to the automated Battle Planning System (BPS)**.**

5. Go-To- War Operational Enhancements: ESS in support of operational enhancements makes the technology more affordable than a single training enhancement system. An expanded discussion of ES uses for the AAN can be found in reference 2.

a. Situational awareness
b. Battlefield visualization
c. Mission planning/rehearsal
d. Course of action analysis
e. Critical decision making.
f. Command & Control (staff uses)
g. Information overload reduction (Info Filtering Tool)

(a) Situational Awareness (SA) can be enhanced by an ESS. The rapid processing and sharing of enemy and friendly location information in a structured format can assist the commander with making timely decisions. ESS can be used to automate the Tactical Decision Making Process (TDMP) because the computer can collect and compile essential enemy information and filter out non-essential information and display as either a 2D or 3D view. The simulation computer can compare old and new enemy situational templates to predict possible enemy actions or intentions. As the enemy closes the computer can display on screen weapon range arcs to alert the crews of their vulnerability to enemy direct or indirect fire. SA will not be degraded by extended distances because UWB technology has the multi-path over the horizon connectivity which can use every vehicle as a store-and - forward relay platform and by operations in a built-up area because UAB is immune to signal interference caused by man-made structures.

(b) Tactical information from the various ground and airborne sensor systems can be ported into the battalion Tactical Operations Center (TOC) for use by the commander and staff to make timely decisions. If necessary this information can be displayed on every vehicle tactical display instantaneously to give every crew a clear picture of enemy action. The rapid graphic display of enemy info like a Family of Scatterable Mines (FASCAM) minefield becomes a powerful tool that can save time and lives. Operations orders and graphics can be transmitted electronically and thereby reducing report preparation and distribution times.

(c) Mission planning and rehearsal can be realistically accomplished by conducting a virtual reconnaissance of the battle area or a virtual look back at the defensive position, and a virtual rehearsing against a GGF on the same terrain database and using similar light & environmental conditions. Electronic planning and stealth reconnaissance will maximize the use of planning time and minimize exposure to enemy observation and fire. Because of the inherent covertness of UWB technology, critical information can be passed freely during both planning and rehearsal phases. Stetliminating some of the unknowns and reinforcing proper execution while rehearsing enemy "what if" situations. The concept of "Perfect practice makes for perfect execution" would enhance crew confidence.

(d) Developing the best course of action can be made easier by running the various Blue courses of action virtually against the Red courses of action. Quick simulations can be run to determine possible results of the various courses of action. The commander can make his final decision based upon the result of the computer comparative analysis and risks involved.

(e) An ESS can automate the collection & dissemination of key information (SA) automatically, thereby allowing rapid decision making based upon the most recent information available and models of enemy tactics, techniques, procedures and order of battle information. Using the ESS to reduce the commander's information processing duties can abate the stresses of combat decision making. ESS contributes to digitization as a force multiplier.

(f) Command, control and communication will be expedited and improved by using the on-board processing capacity, smart models, intelligent agents, covert digitized communications, and the real time display of enemy and friendly activity / status. Graphical displays vice verbal transmission of critical information will save time and standardize information exchange. Commanders and staffs can overwatch unit personnel & logistical status and anticipate support requirements.

(g) ES can be used to perform as an intelligent agent to filter out extraneous information and provide non-redundant transmission of information that is crucial to decision making. The resultant filtered output to the human decision-maker will permit faster and more accurate decisions and prevent information overload and display clutter. The system can be embedded into the Advanced Tactical Command & Control System (ATCCS) and the display tailored to show information the commander considers critical to his decision making process.

Embedded System (ES) Architecture

a. Three architecture approaches are currently being studied.

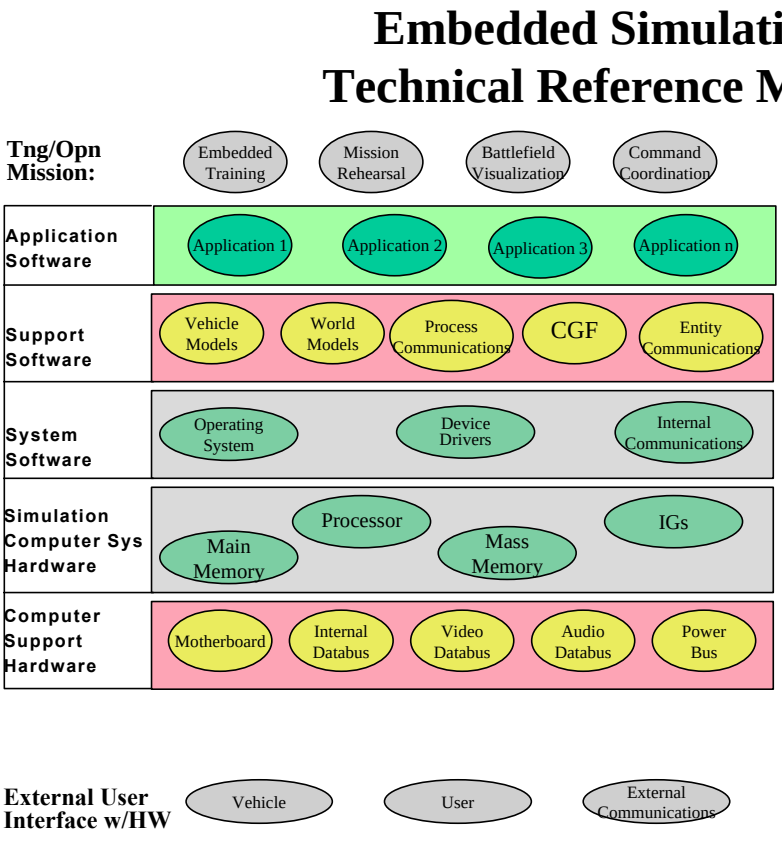
(1) A completely separate stand-alone ES subsystem with its own processing image generator and SW. Advantages would be utilizing off-the-shelf and ruggedized HW, lower cost, use of state of the art graphics cards and processors and least interference with actual vehicle HW & SW. Disadvantage is the need for additional space requirements.

(2) Fully embed ES into the vehicle subsystems by adding or upgrading the vehicle's own computer and SW architecture to accommodate the additional requirements to support ES. Advantages would be not requiring any extra space and use of the current video interface. Disadvantage is that new components would have to be militarized and integrated with the existing system and the associated costs.

(3) A hybrid or combination of 1 and 2. In this approach the idea would be to find the optimal approach that utilizes as much of the vehicle computer and networking capability, but anything new or to costly to militarize is put in a separate subsystem. The disadvantage of this approach is that it will still require extra space.

b. Stet the final decision to implement an architecture approach will be made at a later time. Hardware and software architecture for ESS must be done in a away that will allow for seamless integration in any potential vehicle platform. This will require emphasis on architecture interfaces (HW & SW) that can be fully embedded into the vehicle architecture design. Computer technology and graphic systems are improveing at such a rapid rate that the ES architecture must be designed in a way to reduce upgrade costs. ESS will be developed under an architecture-based approach that will emphasize loosely coupled interfaces and maximum use of commercial standards and software Application Programming Interfaces (APIs). . We can show with the below ES Technical Reference Model the interrelationship between the HW & SW needed for a fully ESS. Future vehicles will be designed with ES as part of the vehicle architecture.

c. **ES Technical Reference Model**



Significant Capabilities

- Embedded Mission Rehearsal
- Embedded Training
- Battlefield Visualization
- Command Coordination
- Virtual Test & Evaluation
- Simulation Based Acqusition
- 3D Mission Visualization
- Stealth / Virtual Recon

Functional Drivers

- Vehicle Simulation Mode (i.e. weapons, mobility, ...)
- Virtual World/Virtual Target Injection
- Mission Planning/Scenario Generation System
- Terrain Database Generation System
- Stealth/Flying Carpet Mode
- Automated Exercise Manager
- After Action Review/Replay System
- Vehicle to Vehicle Simulation Communication Architecture
- Entity Generation System (i.e. ModSAF)

Figure 2

Conclusion

The enabling technologies associated with INVEST-STO are a significant first step to meet the training and operational challenges needed to support the Army After Next (AAN). The force projection army of the next century will have the benefit of an autonomous training system and dual use ESS capable of providing improved SA and other operational enhancements. This capability will give new meaning to the “train as you fight” imperative. Intelligent tutoring systems and a robust on-bpard training support package will ensure that the crews attain and sustain proficiency advantages over any adversary. The mental agility and information dominance gained through Force XXI will spawn the technology enablers that will make an ESS a key component of combat and training readiness in all future crew and command & control systems. INVEST-STO is at the leading edge of these future operational and training capabilities .

REFERENCES

1. Bahr, H., Abate C. and Collins J., "Embedded Simulation for Army Ground Combat Vehicles," 19th I/ITSEC Conference Proceedings, December 1997

2. Abate,C., Bahr, H. and Brabbs,J., Embedded Simulation for the Army After Next, Armor, July-August 1998.

Hubert A. Bahr is a Decorated Vietnam Veteran with 28 years of Federal Service. He received his BS degree in engineering from the University of Oklahoma in 1972 and his Masters Degree in computer engineering from the University of Central Florida in 1994. For the past 18 years he has been involved with instrumented Force on Force Ranges. He is currently the lead engineer for the INVEST STO in the Research and Engineering Directorate of STRICOM. His research interests are in the areas of parallel processing, artificial intelligence, and computer architecture. He is also pursuing his Ph.D. at the University of Central Florida.

Claude W. Abate is a Senior Military Analyst for Sherikon, Inc. and is currently supporting the Simulation Technology Division, Research and Engineering Directorate of STRICOM. He is a graduate of Florida Southern College and has a Masters of Science Degree from Florida State University. Prior to joining Sherikon, Mr. Abate was a career Army officer with a variety of command and staff assignments in the US and overseas. As a retired Colonel, his experience includes a perspective as a training and doctrine developer and Training Brigade Commander at the US Army Armor Center and School and as an opposing force commander at the National Training Center. He has two years experience working with PM CATT on the Close Combat Tactical Trainer and is currently the project coordinator for INVEST-STO. His military schooling includes the Command and General Staff College and the Army War College.

DEVELOPING SYNCHRONIZED PLAYER MODELS FOR EMBEDDED TRAINING

Vanna McHale and Wesley Braudaway Ph.D.
Science Applications International Corporation (SAIC)
Orlando, Florida

Abstract

The Synchronized Player Models (SPM) project supports the U.S. Army Inter-Vehicle Embedded Simulation Technology (INVEST) Science & Technology Objective (STO) Program [1]. The overall goal of the SPM project is to reduce the network bandwidth required to maintain synchronization between a Live vehicle, a Modular Semi-Automated Forces (ModSAF) player model simulation and its associated clone models in separate simulation environments. The SPM project conducted a series of experiments in order to determine the feasibility of the SPM objective. The first experiment, reported in this paper, focused on the ability to have computer-generated forces operate identically in separate simulation environments without requiring network communication. To obtain this level of synchronization it is necessary to have a *repeatable* ModSAF that provides simulation events (e.g., vehicle location events, firing events, damage events) that occur at the same simulation time in each run of the same scenario.

This paper discusses the use of repeatability to support synchronized embedded simulation and focuses on the modifications required to produce a deterministic, repeatable ModSAF. Experiments were conducted to test and demonstrate the repeatable ModSAF and are illustrated in this paper. These ModSAF modifications, that were developed in support of SPM, were the basis for developing the repeatability mode currently supported in the ModSAF version 4.0 baseline.

Authors Biography

Vanna McHale is a member of the Advanced Simulation Research Team within SAIC's Orlando Operation and a scientist on the Synchronized Player Models project. Ms. McHale received her B.S. in Computer Science from the University of West Florida and has been actively involved in the M&S community since 1992.

Wesley Braudaway, Ph.D. is a member and technical lead of the Advanced Simulation Research Team within SAIC Orlando's Operation. Dr. Braudaway was the Principal Investigator for the Synchronized Player Models project. He was also the System Architect for CCTT SAF and has been involved in several Computer Generated Forces related research and development projects. Dr. Braudaway received his Ph.D. from Rutgers University's Computer Science Department and has been actively involved in the M&S community since 1991.

DEVELOPING SYNCHRONIZED PLAYER MODELS FOR EMBEDDED TRAINING

Vanna McHale and Wesley Braudaway Ph.D.
Science Applications International Corporation (SAIC)
Orlando, Florida

1. INTRODUCTION

Simulation technology advances can be leveraged into a form suitable for embedding into ground vehicles for training, mission planning, and other operational uses. Embedded Training (ET) is a capability designed into or added onto operational hardware and software systems that enables it to provide the simulation cues necessary to train crewmembers. This on-board technology will allow mission rehearsal and sustainment training to occur whether the soldiers are at home stations or deployed.

As part of this simulation environment, collective operation requires the synchronization of multiple embedded simulations. Using today's distributed interactive simulation technology, the volume of data transfer required to support embedded training at the unit and battalion level is a significant obstacle because of the network and communication limitations of the fielded systems. Typically, the fielded systems rely on wireless communication, which provide very low bandwidth for simulation use. The Simulation, Training and Instrumentation Command (STRICOM) is conducting the Inter-Vehicle Embedded Simulation Technology (INVEST) Science and Technology Objective (STO) to address the technologies required to provide this ET capability [1].

Providing deterministic Computer Generated Forces (CGF) as part of the simulation environment solves part of the ET problem by removing the need for communication to synchronize the computer-generated models. Suppose each simulation environment has its own deterministic CGF to simulate all computer-generated models. The simulation environments will produce exactly the same simulation event sequence for the same scenario without requiring any coordination. If the input to these CGFs is synchronized then there is no need to synchronize the computer-generated parts of the simulation environments as done today using the Distributed Interactive Simulation (DIS) protocol.

The SPM project chose as its simulation platform the Modular Semi-Automated Forces (ModSAF) system. This paper describes the necessary modifications to implement a deterministic or *repeatable* ModSAF. This paper is organized to describe the synchronization challenge for embedded simulation, the solution to synchronization using a repeatable CGF, the effort required to make ModSAF repeatable, and the remaining effort required to complete the synchronization of multiple embedded simulations.

2. PROBLEM DEFINITION

The INVEST objective is to provide a simulation environment for both individual vehicle operations and multiple vehicles that interoperate within a collective simulation. In the collective mode, the simulation environments (one for each live vehicle) must be synchronized to present an identical synthetic situation to each vehicle concurrently.

There are two types of synchronization required for this modeling (see Figure 1).

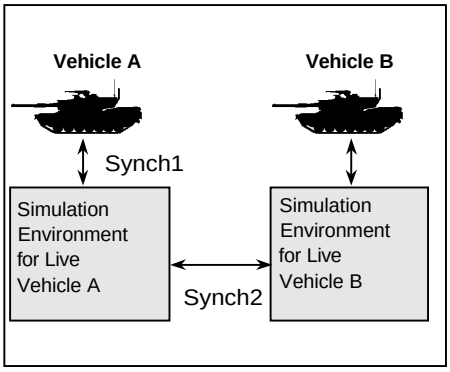


Figure 1: SPM Synchronization

This first synchronization activity occurs between the live Vehicle A and its simulation environment. The interaction of the vehicle and its simulation environment is achieved by a direct coupling of the vehicle's vision blocks and controls to the simulation environment. A virtual model in the simulation environment representing the live vehicle, called the player model, replicates the behaviors of an actual vehicle under the command and control of its crew. Any interaction between the live vehicle and simulated entities is achieved as a side effect of the interaction between the player model for the vehicle and the simulated entities within the simulation environment.

The second synchronization occurs between Vehicle A's simulation environment and Vehicle B's simulation environment. Contained in the simulation environment are CGF (vehicles, units, munitions, etc.) and a player model representing each live vehicle participating in the collective simulation. The synchronization activity makes each environment identical as defined by state data and events affecting each virtual entity regardless of whether they are computer-generated or player models. For example, in addition to replicating live vehicle A's simulation environment, a "clone" of that player model must exist within vehicle B's simulation environment that replicates that same behavior. This synchronization is implemented today using the DIS protocol and communication channels requiring very high bandwidths. However, in meeting the objectives of INVEST STO, high bandwidth and physical connectivity are not feasible alternatives.

3. SYNCHRONIZATION THROUGH REPEATABILITY

The CGF of an embedded simulation can be synchronized by ensuring that each simulation environment is started at the same time and that the computer-generated models process the same events with respect to time in each simulation environment. Assuming that all other aspects of the simulation environments are synchronized, synchronization of the computer-generated models is achieved using a repeatable implementation of the CGF in each environment. Each CGF replicates the exact simulation of all computer-generated models in each simulation environment.

A CGF implementation is *repeatable* if the simulation events (e.g., vehicle location events, firing events, damage events) occur at the same simulation time in each run of the same scenario (where "scenario" is defined as a set of initial conditions and external events). By satisfying this requirement, the CGF implementation within each embedded simulation environment will result in a synchronized simulation environment as long as the initial conditions are the same, the simulation time is synchronized and the external events are synchronized. No additional communication will be necessary to synchronize the CGF models in each simulation environment.

4. IMPLEMENTING ModSAF REPEATABILITY

Although CGF implementations, such as ModSAF, are not typically designed to be repeatable, they can be modified to provide a repeatable mode of execution. Repeatability can be achieved by modifying the scheduling of simulation events, the generation of stochastic events, and the elimination or control of distributed events.

4.1 Event Scheduling

Many CGF implementations, including ModSAF, are event simulations that are managed to ensure a perceived real-time performance. They are designed to emulate a real vehicle by implementing a model that performs as close to the vehicle's actual real-time performance as possible.

A model's behavior is implemented as discrete event changes maintained in an event queue, such as changing its location over time, firing weapons, or taking damage. Each event in the queue is executed relative to a simulation time that is also updated periodically with respect to real-time. An attempt is made to maintain a simulation time equivalent to real-time so that the model's performance appears to correctly emulate an actual vehicle's real-time performance. This differs slightly from discrete event simulations where events in the queue are executed at an explicit, predetermined simulation time.

While the generation and execution of simulation events are deterministic and repeatable, the synchronization of simulation time to real-time is

not repeatable. As the operating system interrupts the simulation to make system service calls at different times and for different periods from simulation run to simulation run, the execution of events with respect to real-time (relative to the start of the simulation) will also vary. Because the discrete events are a sampling of continuous real-time and therefore an approximation of real-time, the outcome of an event may vary if a different sampling occurs between the runs. For example, consider the same movement event for a vehicle in two different simulation runs (as shown in Figure 2).

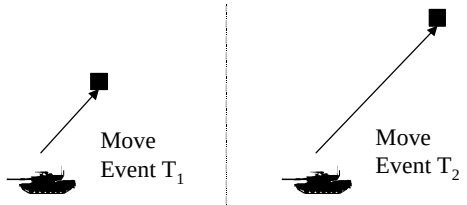


Figure 2: Same Discrete Event in different SRs

Because real-time advanced further in SR2 (possibly due to a longer system interrupt), the vehicle traveled farther during this event in SR2 than in SR1 in order to maintain the real-time approximation. Suppose that this update in both runs placed the vehicle within line of sight of an opposing vehicle and that opposing vehicle reacted immediately by firing its weapon. Because this observation and reaction occurred at different simulation times within the two simulation runs, the executions are no longer identical (i.e., not repeated).

Because of the great dependency between events, as a single event between the simulation runs diverges with respect to time, the differences between the runs will cascade until there are significant variations in the simulation outcomes of each.

To provide a repeatable mode for ModSAF, its event scheduling mechanism was modified. These modifications included changes to the clock and scheduler implementations, and the scheduling of behavior models.

4.1.1 Clock Implementation

To support real-time simulation, the ModSAF real-time clock and its simulation clock were tightly interconnected. As stated earlier, when running in a real-time mode, the simulation clock is dependent on the real-time clock and is advanced with respect to the real-time clock. However, what was unusual about the ModSAF implementation was that the procedure to advance the simulation clock also always advanced the real-time clock. The ModSAF clock implementation was modified to remove this reverse dependency.

In ModSAF's repeatable mode, ModSAF's real-time clock continues to be based upon the system clock. However, the simulation clock was modified to advance to the next event's time on the event queue after processing each event. The continuous frame update rate for the simulation clock was disabled, essentially severing the simulation clock to real-time clock dependency.

4.1.2 Scheduler Implementation

A thorough review of the ModSAF scheduler was conducted to determine areas resulting in random scheduling and/or invoking of events and function calls. ModSAF's scheduler implements four queues: a high priority real-time queue, a periodic real-time queue, a deferred real-time queue, and a simulation time queue. All four queues utilize the real-time clock to invoke events on the queues. The simulation queue implementation was altered to be event driven and based upon the simulation clock rather than the real-time clock to produce a repeatable ModSAF mode. Therefore, when a simulation queue's event is invoked in this mode, the simulation clock is advanced to the time of the next event on the simulation queue.

4.1.3 Scheduling Behavior Models

To utilize the modified, event-driven simulation clock, the scheduling of behavior and physical model update functions was moved from the real-time queue to the simulation queue. All non-simulation functions such as user interface and network functions remained on the real-time queues.

The simulation processing of each vehicle occurs in that vehicle's "tick" or update function. All vehicle events are generated as part of this tick

and therefore, have the biggest influence on repeatable performance. Moving this function and its associated unit behaviors to the simulation queue will provide the required ModSAF repeatability by removing them from the influences of the real-time clock. No other modification to the models or the modeling architecture was required.

4.2 Stochastic Events

Many CGF systems, including ModSAF, include an implementation of stochastic events. For example, given a vehicle is hit by munitions, it may lose its ability to move, it may lose its ability to shoot, or it may be totally destroyed depending of some statistical representation of the chances for each alternative. These events are implemented using algorithms that are based on a number taken from a random number sequence. A repeatable random number sequence is easily implemented by using the same random number seed between simulation runs.

ModSAF's random number seed was not initialized in one central location and was corrected as part of this effort. ModSAF was modified to provide initialization within the random number generator library to provide a constant random number seed for ModSAF's repeatable mode.

4.3 Distributed Events

The distribution of simulation events on a network also impacts repeatability since it is difficult to guarantee the execution time of delivered events between simulation executions. This is due to network latency, variable order of network packets, and the variability in times at which the operating system services its network operations. A repeatable CGF could be achieved either by not allowing distributed simulation events or by explicitly controlling the network events. Controlled network events can be achieved using a reliable network mechanism and some reliable time management capability that alleviates the problem of network latency and system network management.

In ModSAF's repeatable mode, this problem is avoided by only providing repeatability in a non-networked mode.

5. MODSAF REPEATABILITY EXPERIMENT

ModSAF's repeatability was confirmed by experimentation with several scenarios to demonstrate that ModSAF's repeatable mode generates a duplicate simulation outcome for the same scenario. For a particular scenario, two separate ModSAF executions were run to collect data. The data was collected to confirm that identical location update, fire, and damage events occurred at exactly the same simulation time in successive runs. Several other scenarios were used to determine the success of the repeatable ModSAF implementation relative to a variety of behaviors. Data collection and analysis was performed for the following areas:

Scheduler Analysis to determine which function calls were placed on the scheduler, the number of calls made to the functions, and the cumulative processing time.

Queue Scheduling to identify the function scheduling sequence for the deferred, periodic and high priority real-time queues and scheduled simulation time queue.

Random Number Generator to analyze the random number generator activity, to determine its calling functions and to determine the calling event simulation time.

Vehicle Specific Data to gather the vehicle's position (x and y coordinates) and to obtain its damage assessment with respect to simulation time.

Using this data collection, graphs were created to represent and assess ModSAF's repeatable performance. The first graph (see Figure 3) compares the position variation for the same vehicle between two runs of the non-repeatable ModSAF. Position variation is defined as the distance between two instances of the same vehicle at the same simulation times between the two runs of the same scenario. This graph shows the cascading effect of event differences over time between two runs.

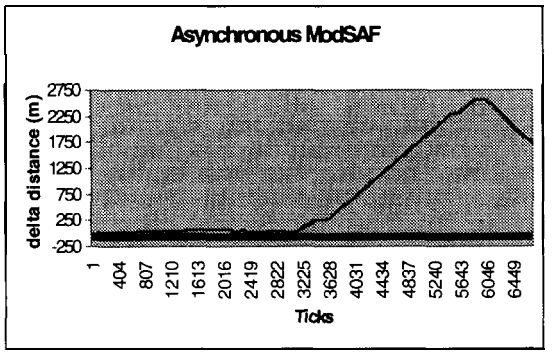


Figure 3: Vehicle Location Difference, Non-Repeatable

The second graph (see Figure 4) shows the same change in position for the same vehicle from one run to another using ModSAF's repeatable mode. This graph illustrates that the vehicle's position throughout the entire scenario remains consistent from run to run (i.e, the distance between the vehicles instances within each run is always zero).

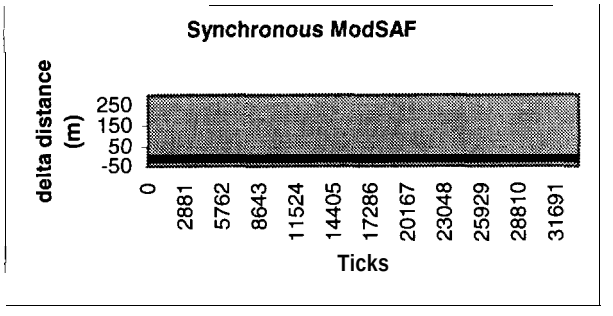


Figure 4: Vehicle Location Difference, Repeatable

In addition to location data, damage assessment data was also analyzed to determine repeatability of fire and detonation events and overall vehicle damage. The third graph (see Figure 5) shows the damage assessment data collected for all eleven vehicles within the scenario. Each run is illustrated as a separate series to show that within the non-repeatable ModSAF, no two events happen in the same manner at the same simulation time (time events in the graph).

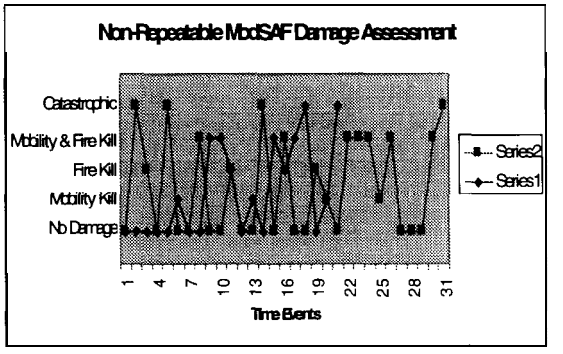


Figure 5: ModSAF Damage Assessment, Non-Repeatable

The final graph (see Figure 6) represents the damage assessment data using the ModSAF repeatable mode. In this figure, the two runs are individually graphed, showing that run 1 is entirely overlaid with the data from run 2. This illustrates that over the entire scenario, all vehicles received the same damage at the same scenario time, from one run to another.

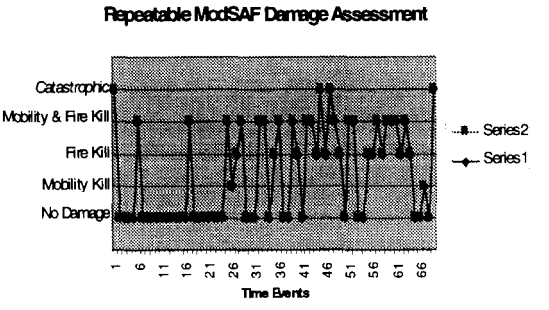


Figure 6: ModSAF Damage Assessment, Repeatable

Numerous other scenarios were executed to test the robustness of the ModSAF repeatability mode. These scenarios incorporated a variety of additional behaviors and obstacles. In each case data collected about the simulation events was consistent from one run to the other, ensuring that repeatability was attained within ModSAF.

6. FURTHER SPM EFFORT

The remaining SPM project goals [2] address the synchronization between a live vehicle and its player model, and the communication of behavior control updates. Current SPM activities are

focusing on continued experimentation to refine the understanding of the player model control interface, behavior parameterization, and overall bandwidth issues. Additional experiments are being developed to investigate the capability to synchronize the simulated models using the state changes of vehicle behavior parameters rather than vehicle location updates. It is believed that this method of synchronization combined with a repeatable CGF will substantially reduce the bandwidth required to synchronize a distributed simulation of player models.

In future phases, additional experiments will be performed to identify and implement the control between ModSAF and the behavior control interface based upon the INVEST STO operations identified by the INVEST Architecture Working Group.

The ongoing SPM experimentation will also consider alternative behavior control strategies within ModSAF to control the synchronization of player model and its associated clone models.

The results of current and future experiments will influence the final SPM architecture and implementation. It is intended that the SPM prototype will be integrated into a developed ET architecture whose objective is to demonstrate the viability of the embedded simulation approach.

7. CONCLUSIONS

The objective of the SPM architecture is to provide Computer Generated Forces that will interact with live vehicles through use of a player/clone model approach. This objective must be achieved while using a lower bandwidth than the current DIS protocol approach for implementing simulation environment synchronization. To achieve this objective, it is believed that dead reckoning at the behavior abstraction level and reducing the requirement for communication to achieve synchronization will reduce the bandwidth [2].

Through the use of a repeatable CGF, SPM achieves synchronization of the computer-generated models in different simulation environments without resorting to DIS protocol. This approach assumes that the scenario (inputs and distributed events) implemented in each environment is synchronized. A repeatable CGF is achieved by scheduling the simulation events independent of real-time, generating stochastic events from a fixed random number sequence, and either eliminating or control the processing of distributed events. The results of this effort were fully integrated into ModSAF version 4.0 to provide a ModSAF with a repeatable mode.

8. ACKNOWLEDGEMENT

The authors would like to acknowledge the efforts of Deanna Nocera, Gene McCulley and Eddie Cason for their contribution to this analysis and repeatable ModSAF development.

The authors would also like to acknowledge the efforts of Derrick Franceschini and Gene McCulley of the Advanced Distributed Simulation Technology (ADST - II) program for their complete integration and testing of the repeatable mode into the Army's ModSAF version 4.0 baseline.

The work presented in this paper was sponsored by STRICOM under contract N61339-97-C-0040.

9. REFERENCES

[1] Bahr, H.A., "Embedded Simulation for Ground Vehicles," Spring 97 Simulation Interoperability Standards Workshop Proceedings, Institute for Simulation & Training, Orlando, FL, March 1997.

[2] Braudaway, W., and Nocera, D.L., "Synchronized Player Models for Embedded Training," Spring 98 Simulation Interoperability Standards Workshop Proceedings, Institute for Simulation & Training, Orlando, FL, March 1998.

BEHAVIOR MODELING FRAMEWORK FOR EMBEDDED SIMULATION

Amy Henninger, William Gerber, Ronald DeMara, Michael Georgiopoulos, and Avelino Gonzalez
University of Central Florida
Orlando, FL.

ABSTRACT

Although embedded training has become the preferred approach for training military forces, it is surrounded by a variety of technical challenges. The Inter-Vehicle Embedded Science and Technology (INVEST) Science and Technology Objective (STO) program explores technologies required to embed simulation in combat vehicles. One of these requirements is to provide a simulation environment in which computer generated forces, manned simulators, and live vehicles may interact in real-time. Unfortunately, providing this geographically distributed and untethered real-time interaction is severely limited by the communications requirements imposed by the need to convey large amounts of data between the respective players. By extending the concept of Distributed Interactive Simulation (DIS) dead-reckoning, a vehicle movement method, to the behavioral level, this limitation may be mitigated. The Vehicle Model Generation and Optimization for Embedded Simulation (VMGOES) project at the University of Central Florida is focusing on this aspect of the INVEST program. This paper presents the specifications and development process of VMGOES.

ABOUT THE AUTHORS

Amy Henninger is a doctoral candidate in computer engineering at the University of Central Florida, a Research Fellow at U.S. Army STRICOM, and a recipient of the Ninth Annual I/ITSEC Scholarship. She has earned B.S. degrees in Psychology, Industrial Engineering, and Mathematics from Southern Illinois University, an M.S. in Engineering Management from Florida Institute of Technology, and an M.S. in Computer Engineering from UCF.

William Gerber, Lt. Col., U.S.A.F. (Ret.), is a Ph.D. student in computer engineering at the University of Central Florida and a Research Fellow at U.S. Army STRICOM. He has a B.S.E.S. degree in Astronautics and Engineering Sciences from the U.S.A.F. Academy, an M.S.E. in Nuclear Engineering from the University of California at Los Angeles and an M.S.Cp.E. in Knowledge-Based Systems from UCF.

Ronald DeMara is a full-time faculty member in the Electrical and Computer Engineering Department at the University of Central Florida. Dr. DeMara received the B.S.E.E.. degree from Lehigh University in 1987, the M.S.E.E. degree from the University of Maryland, College Park in 1989, and the Ph.D. degree in Computer Engineering from the University of Southern California, Los Angeles in 1992.

Michael Georgiopoulos is an Associate Professor at the Department of Electrical and Computer Engineering of the UCF. His research interests lie in the areas of neural networks, fuzzy logic and genetic algorithms and the applications of these technologies in cognitive modeling, signal processing and electromagnetics. He has published over a hundred papers in scientific journals and conferences.

Avelino Gonzalez received his bachelor's and master's degrees in Electrical Engineering from the University of Miami, in 1973 and 1974, respectively. He obtained his Ph.D. degree from the University of Pittsburg in 1979, also in Electrical Engineering. He is currently a professor in the Electrical and Computer Engineering Department at UCF, specializing in human behavior representation.

BEHAVIOR MODELING FRAMEWORK FOR EMBEDDED SIMULATION

Amy Henninger, William Gerber, Ronald DeMara, Michael Georgiopoulos, and Avelino Gonzalez
University of Central Florida
Orlando, Florida

INTRODUCTION

The combination of computer simulation and networking technologies has provided the U.S. military forces with an effective means of training through the use of Distributed Interactive Simulation (DIS). DIS is an architecture for building large-scale simulation models from a set of independent simulator nodes (Smith, 1992) that represent one or more entities in the battlefield simulation. By communicating over a network via a common protocol, these entities are able to exist simultaneously and interact meaningfully in the same virtual environment. Currently, however, the ability of live vehicles to interact with these simulated forces in the virtual world is constrained by the communication requirements needed for real-time interoperability. Eliminating or reducing this impediment would enhance military training in a number of ways. For example, it would diminish the costs associated with having live vehicles travel to maneuver ranges for live exercises. Also, by shifting more of the training to operational units, it would reduce the costs associated with the training schools. In essence, the military could rely less on formal school-house training, more on deployable training systems, and fundamentally make training more readily available on an "as-needed" basis.

To accomplish these objectives, the Department of Defense has recently initiated an effort to determine how embedded training and advanced simulation technologies could be used to overcome the obstacles surrounding this technology. One problem, for instance, is that in order for a driver of a live vehicle to train in a virtual domain, he must be able to traverse the artificial/virtual terrain. Correspondingly, he must be able to see the other live and virtual entities on the virtual battlefield and interact with them in real time. To accomplish this, the embedded training systems must sustain the transfer of massive volumes of data. Unfortunately, the networking and communications limitations of currently fielded systems make the transfer of this data using current DIS supported techniques a strenuous task.

Current forms of DIS dead-reckoning are viewed as vehicle movement methods that are used to reduce DIS packet traffic. By communicating a given vehicle's location, velocity and acceleration to other DIS simulators, the models residing on these simulators can predict the unperturbed near term physical location of the vehicle. In the event that this vehicle begins to deviate from its predicted path, the simulator responsible for creating the entity will send out an update of the vehicle's true location to the other simulators. Thus, the predictive utility of the dead-reckoning model is pivotal to the success of network traffic minimization.

The requirement to transfer enormous volumes of data coupled with the communication limitations of currently fielded systems makes using currently existing DIS methods an inadequate approach. Bahr and DeMara (1996) suggest that extending the concept of DIS dead reckoning to the behavioral level may reduce DIS traffic more than merely applying DIS dead reckoning to vehicle movement tasks. Figure 1 illustrates the DIS dead reckoning concept applied to embedded training and simulation. As indicated by Figure 1, this concept requires the distributed processing of multiple vehicle models because every live or simulated vehicle is represented by a model and every model is resident on every vehicle. The vehicle model (VM) serves to predict the actions of the vehicle it represents. When the actions of the vehicle are consistent with the actions predicted by the vehicle's model, all of the copies of that vehicle's model are correctly reflecting the live vehicle's actions. In this instance, the interaction between the other vehicles and the vehicle model in the virtual world is an accurate representation of the vehicles' interactions in the real world. However, if the actions of the vehicle are not consistent with the actions predicted by the vehicle's model, the copies of that vehicle's model

are not correctly reflecting the live vehicle's actions. In this instance, the interaction between the other vehicles and the vehicle model is not consistent with their real world interaction.

As indicated in Figure 1, a system that extends the DIS dead-reckoning concept to the behavioral level requires the identification of discrepancies between the behavior of an actual vehicle and that vehicle's model. The portion of this system that identifies and classifies these discrepancies is referred to as the Difference Analysis Engine (DAE) in Figure 1. By comparing the state of the vehicle model with the state of the actual entity, the DAE identifies whether discrepancies in the behavior as well as the position exist. If there are discrepancies, the DAE determines whether an update is necessary and what that update should be. The types of information provided by the DAE are specified in a future section of this paper.

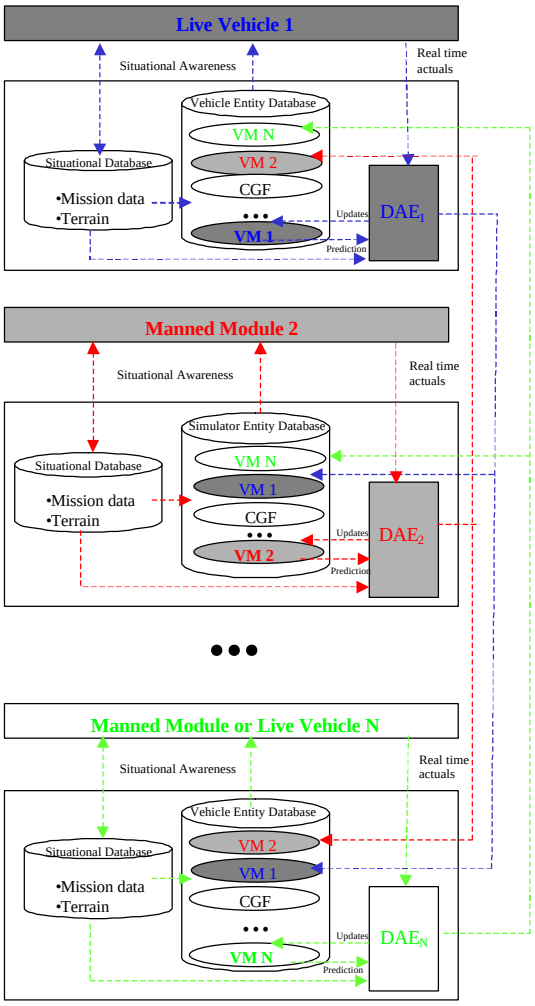


Figure 1. DIS Dead-Reckoning Approach Extended to Behavioral Level

This paper offers a framework for the development of the VM and the DAE. Also, this paper addresses the integration of the two components into the full system known as Vehicle Model Generation and Optimization for Embedded Simulation (VMGOES).

SCOPE OF MODEL

Frequently, DIS simulations use computer controlled combatants known as Computer Generated Forces (CGFs) to populate the battlefield. The behavior of a CGF may be generated by a human operator assisted by software, in which case the class of CGF is referred to as a semi-automated force (SAF) or generated completely by software, in which case the class of CGF is referred to as an autonomous force (AF). The behaviors generated by CGFs are based on doctrine and represent a wide variety of tasks with a reasonable level of detail. Because these behavioral models are fashioned entirely by doctrine, they emulate standard procedures that are acquired from declarative knowledge (i.e., manuals and interviews) and provide a range of *feasible* behavior. However, these models of behavior provide no representation for the 1) implicit knowledge or 2) intrinsic performance characteristics that make "live entities" unique from one another. For example, the current CGF behavioral models used in DIS exercises may simulate the movement of a vehicle to a given location by some standard movement model, but they do not "individualize" that movement method by either assigning or simulating human performance characteristics (e.g., tendency to hug the side of the road, propensity to maintain speed above speed limit, etc.) to it. Thus, behavioral models fashioned entirely by doctrine are often characterized as yielding responses that are "canned", "predictable", or "too perfect". However, the fact that these behaviors are "canned" or "preprogrammed" in no way suggests that these behaviors are simplistic. Prevalent SAF systems have integrated hundreds of thousands of lines of code to successfully emulate the command and control hierarchy of a military unit and its operation on the battlefield. By providing a variety of planned behaviors (e.g., "Conduct a Tactical Road

March", "Attack By Fire", "Service Station Resupply", etc.), situational awareness and assessment, and reactive behaviors (e.g., "Breach a Minefield", "Call for Indirect Fire", "Actions on Contact", etc.), they have successfully provided suitable friendly and enemy forces to populate the battlefield.

The models to be used in this project are conceptually similar to CGF models, but they are distinguishable by the addition of human performance characteristics in the model. In other words, whereas a CGF may *emulate* the selection of a vehicle's cover and concealment position, extending the DIS dead-reckoning concept to the behavioral level requires the *prediction* of the vehicle's actual cover and concealment position. This necessary increase in detail for the VM coupled with the research oriented nature of this project, limits the initial efforts for VMGOES to an exercise smaller in scope than one may find in a typical DIS exercise.

The exercise used in VMGOES centers around a Blufor M1A2 tank platoon or section performing a Tactical Road March and executing an Actions on Contact task in response to a potential enemy threat (i.e., an Opfor T-72 platoon, section, or vehicle). A variety of control parameters can be modified by the VMGOES model users. This allows the users to more fully exercise the model to evaluate its ability to generalize. These parameters are categorized in two groups: (1) task parameters and (2) operational parameters. These parameters and their permissible ranges are defined below.

Task parameters that may be changed by the evaluators are expressed by task. These tasks include Tactical Road March and tasks related to Actions on Contact maneuvers.

Tactical Road March Parameters

Tactical Road March parameters that may be modified include the route and march rate.

Route - may be defined within the constraints of the assumptions/conditions (listed under Assumptions section).

March rate - must be defined within the acceptable limits of the march rates delimited in simulation.

Actions on Contact Parameters

Rules of engagement is the only parameter that may be modified to influence this task.
Rules of engagement - may be initialized as free, tight, or hold to either all or none of the Blufor M1A2 entities.

Operational Parameters

The following operational parameters may be changed in a VMGOES exercise:

Terrain - area where scenario is executed within constraints of the Assumptions section

Blufor Unit Size - tank section or tank platoon

Opfor Unit Size - single vehicle, tank section or tank platoon, and

Opfor Unit Location - positioning (location and direction) of Opfor unit

Assumptions

Lastly, the following conditions/assumptions will apply to the exercises considered by VMGOES:

1. Terrain does not include bodies of water (e.g., lakes, rivers, swamps or ponds).
2. Model does not simulate Command Overrides, Fragmented Orders, or other externally initiated changes in orders.
3. Opfor (T-72 vehicles) operate according to defaulted behavior of simulation unless specified otherwise for a scenario.
4. Blufor units should begin exercise on route, have heading directed towards end of route, and be oriented closely parallel to its position on the route.
5. Manned module always represents the lead tank (i.e., platoon leader).
6. There will be no modifications to terrain (e.g., obstacles or minefields).
7. M1A2 may not initiate calls for support (e.g., indirect fire).
8. The section of terrain east of Barstow Road and west of Hill 720 in the NTC-0101 terrain database will be used for development and tests.
9. The simulation's environmental factors (e.g., weather, tactical smoke, etc.) will not change during a scenario.

10. Tactical Road March tasks may only be assigned to terrain where the road is observable.

MODELING PARADIGM

To develop the vehicle model, VMGOES is using a machine learning technique known as Learning by Observation (Gonzalez, et al, 1998). This technique facilitates the development of intelligent, computational models of human behavior. Although a relatively new concept in the discipline of machine learning, Learning by Observation has been successfully used in a variety of highly complicated, real-world tasks. Pomerlau (1992), for example, used Learning by Observation in the development of an Autonomous Land Vehicle In a Neural Network (ALVINN). In this project, Pomerlau trained a neural network to drive a vehicle through a one-lane road under ideal environmental conditions. Moreover, this network was able to generalize its training to perform satisfactorily in two-lane as well as in dirt roads, and under adverse environmental conditions (snow, rain, etc.). Expanding on this work, Pomerlau et al. (1994) have developed “smart” vehicles as part of Advanced Research Projects Agency's (ARPA's) Unmanned Ground Vehicle (UGV) program, intended to reduce the need for human presence in hazardous situations. These vehicles are capable of driving themselves at speed up to 55 mph for distances of over 90 miles on public roads. Moreover, they are capable of driving both during the day and night, driving on a variety of roads, avoiding obstacles, and even performing parallel parking.

With respect to the vehicle models in this project, Learning by Observation will be used to learn human decision making skills (e.g., reactive transitions, route planning, selection of cover and concealment) and low-level human control strategies (e.g., route following, scanning, etc.). Ultimately, these behaviors will be learned through the observation of a human expert tank commander. However, as a preparatory step, VMGOES is currently developing a VM prototype by learning these behaviors through the observation of a ModSAF M1A2 entity. In other words, instead of using a human as the expert whose behavior is learned, VMGOES is initially using a ModSAF entity as the “expert” whose behavior is learned. This prototype model is referred to as VM_{ModSAF}. It is anticipated that this prototype work will assist in the identification of technical issues that may arise in the second phase of the study and ultimately, will increase the likelihood of successful results in the final system.

VMGOES DEVELOPMENT

In addition to using ModSAF to supply the M1A2 "expert" entity from which VM_{ModSAF} will learn, the ModSAF system is being used to provide the simulated environment in which the vehicle models interact. This allows VMGOES researchers to focus on the development of accurate behavior-based models as opposed to the implementation issues pertaining to vehicle simulation (e.g., physical modeling, weapons modeling, network interface, etc). The VM is embedded in ModSAF and receives input data consistent with sensory information obtainable from the controls and display systems resident in an M1A2. The VM output contains the commands and parameters needed to control the vehicle's motion and weapons' execution. The DAE, alternatively, runs as a separate process and receives input from both the vehicle model and the (live or simulated) master vehicle's interface. In both the VM_{ModSAF} and the vehicle model derived from the human expert (VM_{MM}), this interface supplies sensory information and dead-reckoning type data pertinent to the given model's behavior. This is also true of the interface to the (live or simulated) master vehicle. Once it receives these inputs, the DAE identifies whether discrepancies exist between the vehicle models and the master vehicle and sends out the necessary updates.
The updates sent out by the DAE contain one or more of following four types of information: 1) position, orientation and other basic dead-reckoning information 2) model parameter information, 3) behavior enumeration, or 4) action enumeration. Examples of updates containing these types of information are provided in the following section.

Software Engineering Model

As illustrated in Figure 2, VMGOES has adopted an Incremental Model (Schach, 1993) software engineering process. Using this type of model, the product is designed, implemented, integrated, and tested as a series of thirteen incremental builds, where each build is represented by a numbered circle. Also, each build consists of code pieces that interact together to provide a specific

functional capability. For the most part, the stages down the left-hand column of Figure 2 represent the VM builds and the stages across the bottom two rows represent the DAE builds as they are integrated with the VM. Of the two rows representing the DAE builds, the top row represents the development cycle of VMGOES using a ModSAF entity as the "expert", and the bottom row represents the development cycle of the VMGOES using the human expert. Finally, the integration of the two components (VM and DAE) occurs at the intersection of the column and rows.

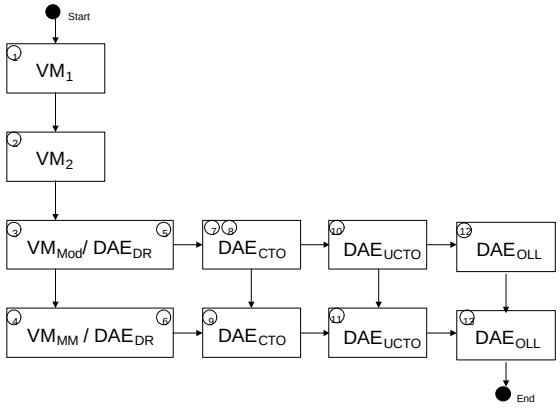


Figure 2. VMGOES Development Model

The thirteen VMGOES model builds presented in Figure 2 are enumerated and defined below according to their functional divisions. Builds 1 and 2 can be described as:

1. Train VM_1 to replicate reactive behavior context transitions by observing a ModSAF M1A2 CGF entity. In this build the reactive behaviors are enumerated as a part of the training set.

2. Train VM_2 to replicate reactive behavior context transitions by observing a ModSAF M1A2 CGF entity. In this build the reactive behaviors are not enumerated as a part of the training set.

In both Builds 1 and 2, a ModSAF M1A2 entity serves as the "expert" from which knowledge is acquired. Also, both models resulting from these builds focus on the acquisition of knowledge pertaining to unit level reactive behaviors. VM_1 and VM_2 are both trained with data containing sensory information (input). The difference, however, is that the output data used to train VM_1 includes the reactive behavior enumeration, whereas the VM_2 model does not have access to this enumeration. As a result, VM_2 must additionally employ some strategy to infer the reactive behavior type that should be associated with a given input vector or cluster of vectors. This second build better reflects the actual task at hand: to learn behaviors from a *human* expert. In other words, since the human expert will not be verbalizing or enumerating his behavior, the methodology for developing the final models in this project must be capable of inferring what that behavior is. It is anticipated that methods developed in Build 2 will assist in meeting this requirement.

Builds 3 and 4 can be described as:

3. Train VM_{ModSAF} to replicate context transitions and actions for VMGOES test scenarios by observing ModSAF M1A2 CGF entity.

4. Train VM_{MM} to replicate context transitions and actions for VMGOES test scenarios by observing a human expert in a M1A2 Manned Module.

The functionality provided by both Builds 3 and 4 is specific to the VMGOES test scenarios as defined in the VMGOES Requirements Document. These are briefly described in the Model Scope section of this paper. The difference between Builds 3 and 4 is that Build 3 uses a ModSAF entity as the "expert", whereas Build 4 uses a human expert. As previously discussed, it is anticipated that modeling strategies learned by the VMGOES team in Build 3 will be useful in the development of the final vehicle model (VM_{MM}) developed in Build 4.

Builds 5 and 6 can be described as:

5. Integrate VM_{ModSAF} with DAE dead-reckoning control (DAE_{DR}) to evaluate VM_{ModSAF}/DAE_{DR} for VMGOES test scenarios.

6. Integrate VM_{MM} with DAE_{DR} to evaluate VM_{MM}/DAE_{DR} for VMGOES test scenarios.

Builds 5 and 6 are simply integration checkpoints in the development cycle. In both of these builds, the vehicle models are being integrated with the

DAEs' basic dead-reckoning control mechanism. This will enable the DAE to update the vehicle model position or orientation with basic dead-reckoning type parameters, in the event that the VM has deviated from the path pursued by the master vehicle.

Builds 7, 8, and 9 can be described as:

7. Train DAE context transition override control (DAE_{CTO}) of VM_{ModSAF} to recognize context transitions for VMGOES test scenarios. In this build, reactive behavior transitions are supplied as part of the training set.

8. Train DAE_{CTO} of VM_{ModSAF} to recognize context transitions for VMGOES test scenarios. In this build, reactive behavior transitions are not supplied as part of the training set.

9. Train DAE_{CTO} context transition override control (DAE_{CTO}) of VM_{MM} to recognize context transitions for VMGOES test scenarios.

In general, these builds focus on the context transition override control of the DAE. This control mechanism allows the DAE to update the VM's enumerated behavior type and is used when the DAE identifies the behavior/context of the live vehicle as being different from the behavior enumerated by the vehicle's model. For example, if the VM is performing a "Withdraw" and the DAE determines that the live vehicle is performing an "Assault", the DAE directs the VM to change its behavior to an Assault.

Specifically, Builds 7 and 8 are using ModSAF as the "expert" and Build 9 is using a human in a manned module as the expert. Additionally, Build 7 and 8, like Builds 1 and 2, are distinguishable by the availability of behavior enumerations in the output of the training set.

Builds 10 and 11 can be described as:

10. Train DAE unrecognized context transition override control (DAE_{UCTO}) of VM_{ModSAF} to recognize unrecognizable context transitions for VMGOES test scenarios.

11. Train DAE_{UCTO} of VM_{MM} to recognize unrecognizable context transitions for VMGOES test scenarios.

Builds 10 and 11 focus on providing the DAE with the capability to completely control the vehicle model, when the DAE is unable to recognize what the live vehicle is doing. Again, Build 10 uses the ModSAF M1A2 entity as the "expert" and Build 11 uses a human expert.

Lastly, Builds 12 and 13 can be described as:

12. Develop procedures for refining previously trained DAE_{UCTO}/VM_{ModSAF} off-line (adjust DAE_{UCTO}/VM_{ModSAF} parameters or contexts) for VMGOES test scenarios.
13. Develop procedures for refining previously trained DAE_{UCTO}/VM_{MM} off-line (adjust DAE_{UCTO}/VM_{MM} parameters or contexts) for VMGOES test scenarios.

Once VMGOES becomes functional, it will have access to more observational data. Those data may help further explain behavior. Builds 12 and 13 capitalize on this fact by providing a mechanism to capture those data and refine the vehicle models.

SUMMARY

This paper described a modeling framework for the development of a system designed to reduce the communications bandwidth required for an inter-vehicle embedded simulation exercise. This system includes a behavior model of the vehicle in the exercise and a difference analysis engine tasked with keeping that model synchronized with its live counterpart. Presently, the vehicle model and the DAE are being developed by using a ModSAF M1A2 entity as the "expert" from which knowledge is acquired. Future endeavors include efforts to apply the lessons learned from this phase of the study to the elicitation of knowledge from a human expert.

ACKNOWLEDGEMENTS

This work was sponsored by the U.S. Army Simulation, Training, and Instrumentation Command as part of the Inter-Vehicle Embedded Simulation and Technology (INVEST) Science and Technology Objective (STO), contract N61339-98-K-0001. That support is gratefully acknowledged.

BIBLIOGRAPHY

Bahr,H.A. and DeMara, R.F., (1996). A Concurrent Model Approach to Reduced Communication in Distributed Simulation, Proceedings of the 15th Annual Workshop on Distributed Interactive Simulation, Orlando, FL.

Gonzalez, A.J., Georgiopoulos, M., DeMara, R.F., Henninger, A, and Gerber, W., (1998). Automating the CGF Model Development and Refinement Process by Observing Expert Behavior in a Simulation. In Proceedings of the 8th Conference in Computer Generated Forces and Behavior Representation, Orlando, FL: University of Central Florida Institute for Simulation and Training.

Pomerlau, D.A., (1992). Neural Network Perception for Mobile Robot Guidance, Ph.D. Dissertation, School of Computer Science, Carnegie Melon University, Pittsburg, PA.

Pomerlau, D., Thorpe, C., Longer, D., Rosenblatt, J.K., and Sukthankar, R., (1994). AVCS Research at Carneghie Mellon University. Proceedings Of Intelligent Vehicle Highway Systems America 1994 Annual Meeting, p. 257-262.

Schach, S.R., (1993). Software Engineering. Aksen Associates Incorporated Publishers, Boston, MA.

Smith, S., and Petty, M. (1992). Controlling Autonomous Behavior in Real-Time Simulation. In Proceedings of the Second Conference in Computer Generated Forces and Behavior Representation. Orlando, FL: University of Central Florida Institute for Simulation and Training.