A Framework for Enabling Virtual Observer Controllers in Synthetic Training

Mandira Hegde, Dan Allison, Todd W. Griffith, Ph.D.

Discovery Machine, Inc.

Williamsport, PA

mhegde@discoverymachine.com, dallison@discoverymachine.com, tgriffith@discoverymachine.com

ABSTRACT

This paper presents a modeling framework to enable the creation of custom virtual observer controllers (VO/Cs) to help naval students meet training objectives. First, we describe an approach to create behavior models to drive the behavior of automated entities used in synthetic training. Next, we describe an approach to create behavior models supporting the functions of intelligent VO/Cs. Specifically, we describe the development of an authoring console and training task blocks used to create custom VO/C architectures as well as the creation of student cognitive process models representative of varying training proficiency levels. We also describe a method to integrate these models into simulations used in synthetic training and a communication architecture supporting communication between VO/Cs and our behavior-driven automated entities. We end with a use case of a custom VO/C running end-to-end in the Joint Semi-Automated Forces™ (JSAF) simulation in a Navy Anti-Submarine Warfare (ASW) training scenario. Although the Navy ASW training domain is the focus of the examples described in this paper, the modeling framework described is not domain specific.

ABOUT THE AUTHORS

Mandira Hegde is a Knowledge Engineer at Discovery Machine, Inc. She graduated from Case Western Reserve University in 2011 with a Bachelor of Science degree in Systems Biology and a minor in Cognitive Science. She has worked on various behavior modeling projects at Discovery Machine, focusing on building deployable knowledge models of intelligent agents for use within synthetic training environments. Her past experience includes neuroimaging research with Case Western, the Cleveland Clinic, and New York University.

Dan Allison is the VP of Professional Services at Discovery Machine, Inc. He has 25 years of experience in system engineering and modeling in a variety of businesses including DoD SONAR and RADAR systems, medical device design and military training systems. His most recent work focuses on knowledge engineering and building deployable knowledge models. Mr. Allison earned a BSEE from Penn State and MSEE from Syracuse University and is six sigma certified. He has received two patents on the topic of deconvolution algorithms and has been an active member of both IEEE and INCOSE.

Dr. Todd W. Griffith is the founder and CTO of Discovery Machine, Inc. He has been working in the area of intelligent systems research for over twenty years and has published papers in the areas of cognitive science, human-computer interaction, and intelligent systems, and has received four patents in the area of knowledge systems. Prior to founding Discovery Machine, Dr. Griffith taught computer science and artificial intelligence at Georgia Institute of Technology and Bucknell University. Dr. Griffith received his Ph.D. in computer science and artificial intelligence in 1999 from Georgia Tech with a focus on modeling the problem solving strategies of Subject Matter Experts (SMEs).

A Framework for Enabling Virtual Observer Controllers in Synthetic Training

Mandira Hegde, Dan Allison, Todd W. Griffith, Ph.D.

Discovery Machine, Inc.

Williamsport, PA

mhegde@discoverymachine.com, dallison@discoverymachine.com, tgriffith@discoverymachine.com

INTRODUCTION

Human observer controllers (O/Cs) traditionally participate in live and synthetic training to determine students' mission readiness. Effective O/Cs are capable of diagnosing student cognitive and affective problems, intervening in training as necessary to help students meet their training objectives, and selecting appropriate instructional treatments to address student problems; without intervention from knowledgeable O/Cs, students may learn poor performance tactics just as well as good performance tactics (Porayska-Pomsta et. al., 2008). However, with only human O/C involvement during training, training effectiveness will vary as a function of the O/C's knowledge and instructional skills. Virtual observer controllers (VO/Cs) can be used to supplement human O/Cs during synthetic training in order to reduce instructor-to-instructor variation. VO/Cs can also reduce instructor workload while increasing instructor presence.

We have identified the following five capabilities essential for effective VO/Cs:
1. VO/Cs must monitor student performance during training and alert instructors to critical student actions in real time.
2. VO/Cs must assess whether students are following reasonable approaches during training. In order to do this, VO/Cs must understand the many ways training tasks can be accomplished in a variety of situations. In addition, VO/Cs must understand that there are different approaches to a single problem and different circumstances under which each approach should be attempted.
3. VO/Cs must identify when to intervene during training. While it is often instructive to allow students to take the wrong path in order to learn from their mistakes, there are other times when the wrong path simply leads to confusion. VO/Cs must recognize this difference and identify when intervention will enhance training.
4. VO/Cs must identify effective ways in which to adapt training scenarios in order to help students.
5. VO/Cs must aid in conducting an After Action Review (AAR). This means summarizing and presenting data relevant to training objectives so that students can clearly understand their own performance.

In this paper, we describe a framework to enable the creation of custom VO/Cs possessing these capabilities for use in synthetic training. Specifically, we describe an approach to create behavior models to drive the behavior of automated entities used in synthetic training, an approach to create behavior models supporting the functions of intelligent VO/Cs, a method to integrate these models into simulations used in synthetic training, and a communication architecture supporting communication between VO/Cs and our behavior-driven automated entities. We end with a use case of a custom VO/C running end-to-end in the Joint Semi-Automated Forces™ (JSAF) simulation in a Navy Anti-Submarine Warfare (ASW) training scenario.

BACKGROUND

Behavior Modeling Approach

Throughout the effort described in this paper, we have used a behavior modeling approach which represents knowledge captured from Subject Matter Experts (SMEs) in a visual language called Task-Method Knowledge (TMK). TMK models are visually represented as hierarchical task networks made up of three types of elements shown in Figure 1: tasks, methods, and procedures. Tasks represent high-level actions to be completed and are depicted in TMK models as rectangular blocks. These tasks decompose into method child elements, depicted as blue ovals, representing ways of accomplishing tasks, and/or procedure child elements, depicted as rectangular blocks

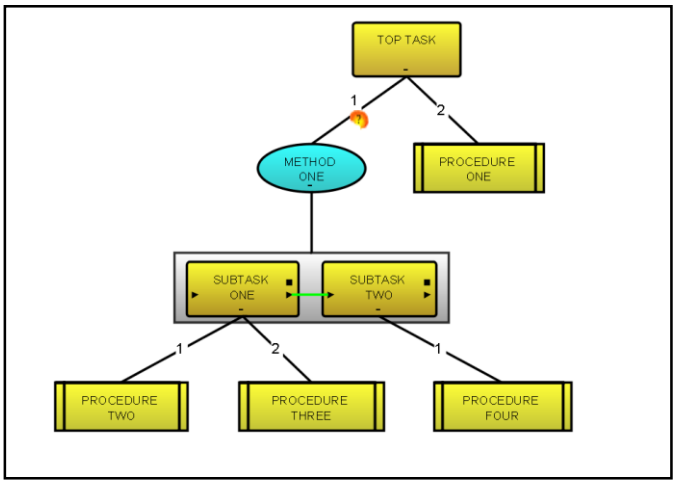


Figure 1. Example TMK Model

with black stripes, containing functions to execute the high-level tasks at runtime. This behavior modeling approach provides many advantages over script or lisp-like representations including more rapid development of new behaviors, transparent behaviors for SMEs during and after development, and transparent behaviors at runtime exposing the decision-making process of entities to operators and easing debugging (Potts *et. al*, 2010).

ARCHITECTURAL OVERVIEW

The general architecture underlying the creation of custom VO/Cs is shown below in Figure 2 and described in the remainder of this section.

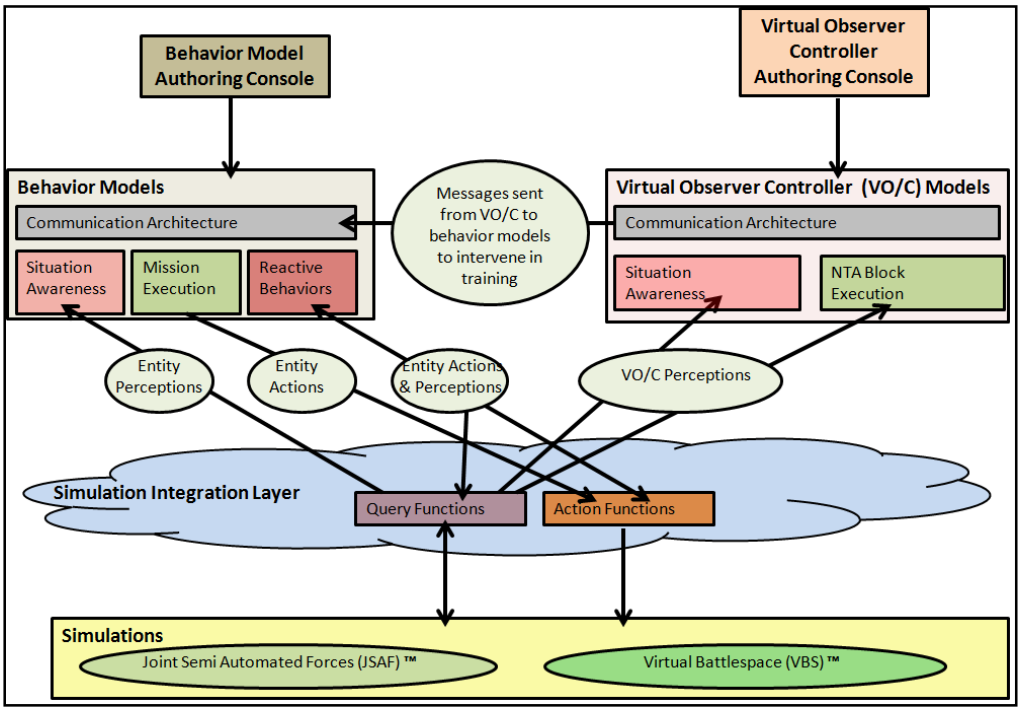


Figure 2. Architecture Enabling VO/Cs

Behavior Model Structure

Using the TMK language, we create behavior models to drive the behaviors of automated entities used in synthetic training. Each of our behavior models are built from a template containing the following branches:

1. Situation awareness processing branch – This branch processes raw data about the entity and its simulation environment and uses it to generate higher level mental models of the world in which it exists. Our research efforts

on previous projects have led to an implementation of Mica Endsley's model of situation awareness within this branch (Endsley, 1995). As such, level 1 data consists of raw sensory perceptions of the entity within the simulation that requires no inference. Examples of this data include the state of the entity itself such as its position, fuel levels, and contacts that appear on its sensors. This raw sensory data provides a basis upon which higher level understanding of the environment can be derived. Level 2 data consists of higher level data inferred from the level 1 data, representing abstract mental models of the simulated environment. Examples of this data might include knowledge about a particular entity that exists in the world, based upon all available sensory data. This knowledge is used as input to the mission and reaction execution branches. The situation awareness processing branch continuously loops throughout the execution of the mission so that entities maintain updated mental models of their surroundings.

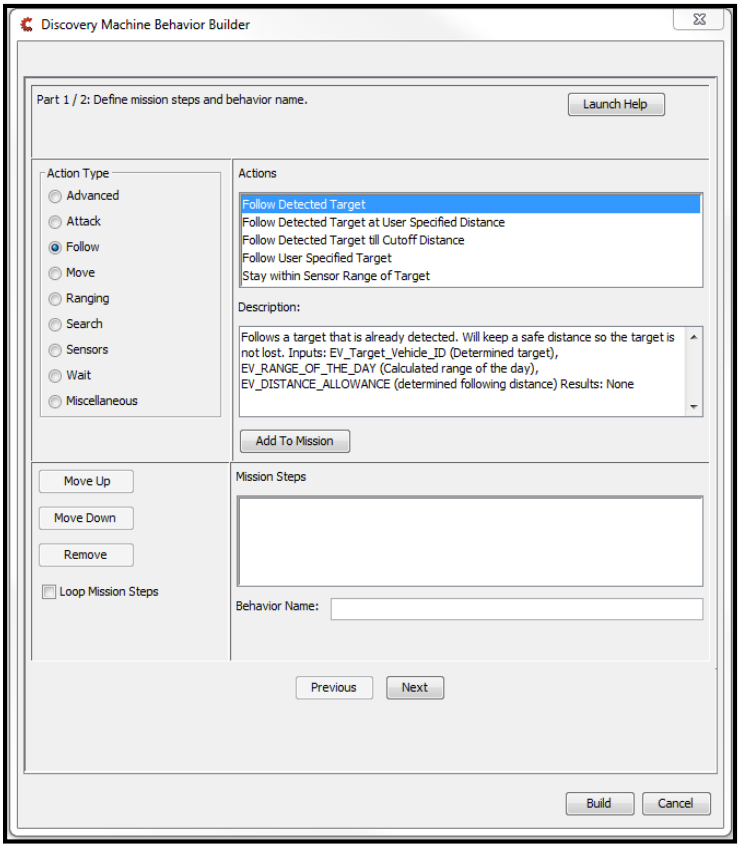


Figure 3. Behavior Model Authoring Console

2. Mission execution branch – This branch consists of a series of instructor-selected basic level actions (BLAs) representing the primary mission to be performed by the behavior. BLAs serve as modular building blocks for the mission, developed to allow instructors to create custom, easily understood missions. The term "basic level" comes from Eleanor Rosch's research which found that humans tend to speak of concepts of actions at certain ontological levels. For example, in telling a story, one would relate actions such as: "got up, brushed teeth, got dressed" as opposed to "got up, walked 15 paces to the bathroom, unscrewed the cap on the toothpaste, etc." (Rosch *et. al.*, 1976). These BLAs, developed at this ontological level, are executed sequentially in the model. On previous projects funded by ONR and NAVAIR-TSD, we have developed BLA libraries for use in a variety of training domains along with authoring consoles to allow users to select their desired BLAs. Figure 3 shows an advisor dialog we have created with BLAs for use in the Anti-Submarine Warfare (ASW) domain.

3. Reaction execution branch –This branch allows the behavior to interrupt execution of the primary mission and react to unscheduled events in the simulation. Development of this branch was necessary as unscheduled events are common during training. For example, when creating a behavior model for use in synthetic training, the instructor may know that the entity tasked with the model will be attacked. However, the instructor will not know at what point during training the attack will take place. If the instructor adds a reaction to handle an attack to the behavior model, the entity tasked with the behavior model will be able to pause its main mission in a timely manner in order to react to the attack. This branch uses a blackboard system combined with listeners to enable the instructor to define layered reactions within a subsumptive architecture.

Figure 4 shows an example behavior model, created using the authoring console partially shown in Figure 3, intended to drive the behavior of an OPFOR submarine within a synthetic ASW training scenario. An entity driven by this behavior model will move on a user-specified route while reacting to requests from either humans or other behavior models to ping, fire torpedoes, and move within range of designated targets. The situation awareness branch in the model is concurrently receiving and perceiving information, processing incoming information, and storing processed information to aid in the execution of the mission (concurrency is depicted by the striped subtask group). The mission is constructed from several BLAs that achieve the mission objective: Move Sub on Route, React to Ping Request, React to Fire Torpedo Request, and React to Move within Range of Target Request.

`

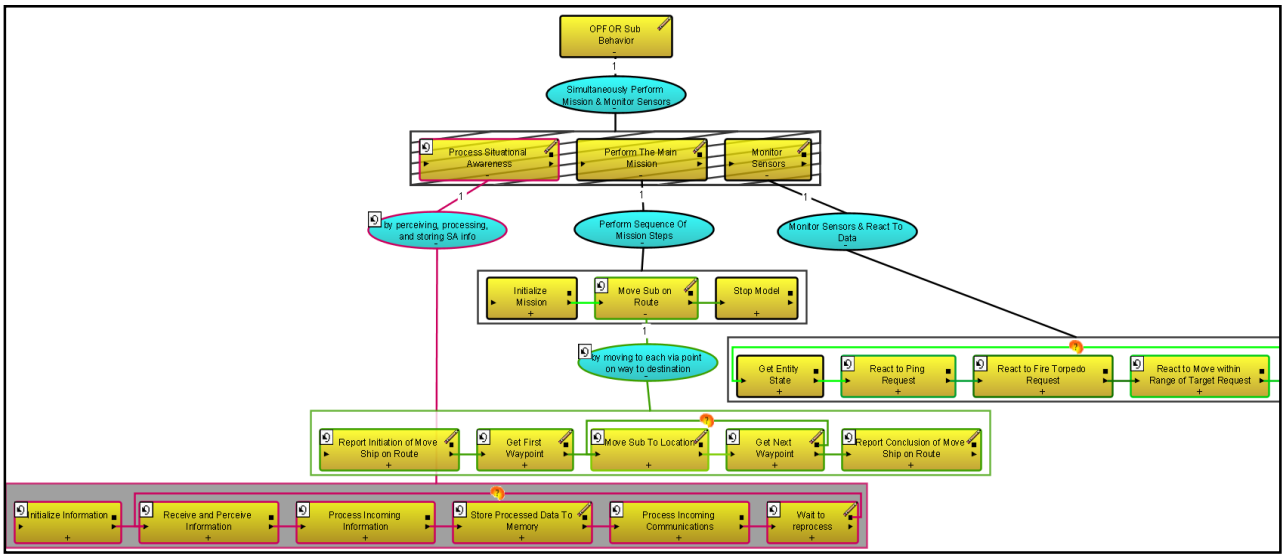


Figure 4. OPFOR Submarine Behavior Model Example

Virtual Observer Controller Model Structure

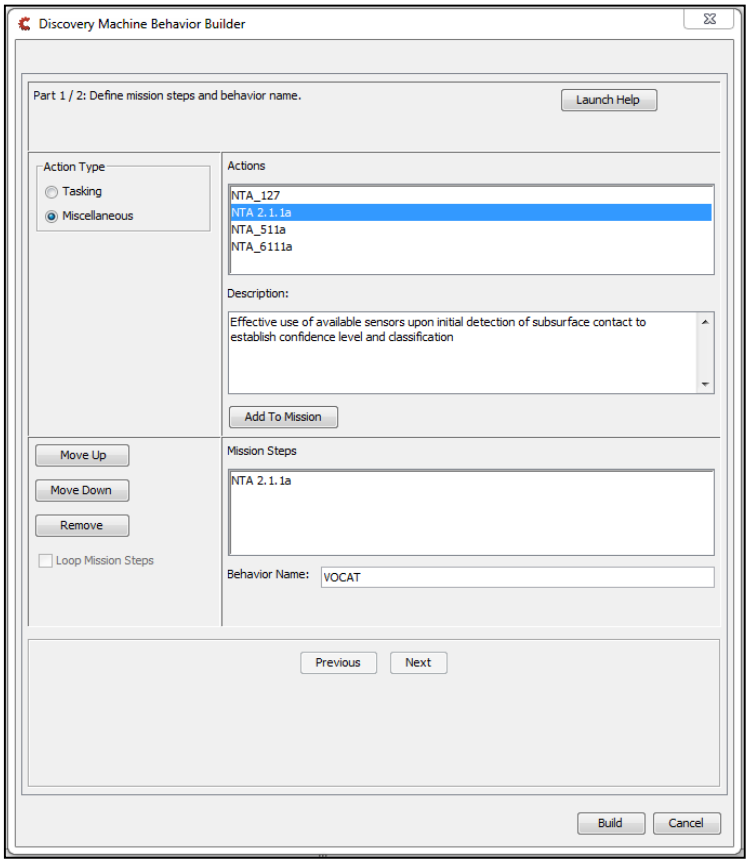
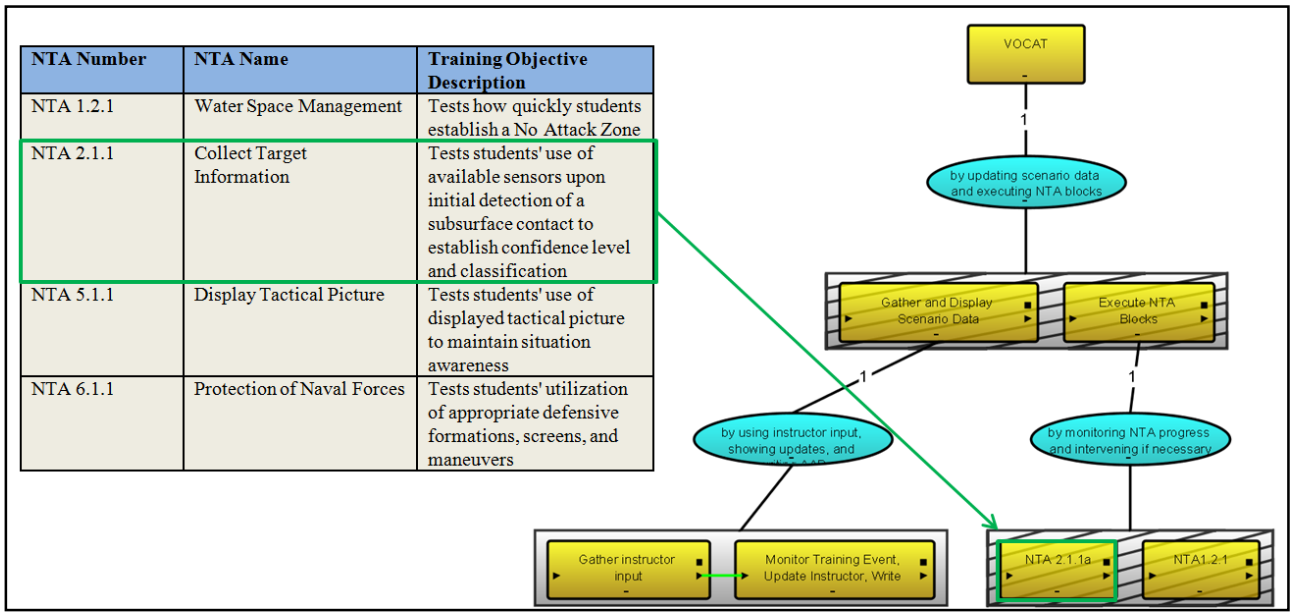


Figure 5. VO/C Authoring Console

Similar to behavior models, VO/C models are built with an authoring console, partially shown in Figure 5, using a TMK template with standard high-level tasks, shown in Figure 6. The first branch, 'Gather and Display Scenario Data,' receives and comprehends input from the instructor and the training environment, performing situation awareness processing. The branch begins to build its situation awareness by gathering key information about the student, such as the student's proficiency at using certain sensors to detect enemies as well as key information about the scenario, such as the amount of time allotted to complete the training mission. Thus, the situation awareness of the VO/C model is similar to the situation awareness of an instructor monitoring a training event. While the training scenario is running, the branch continues to update its situation awareness by monitoring all entities in the simulation and gathering data necessary for AAR. Such data may include entity positions and their relative distances, elapsed time, and student sensor use.

The second branch, 'Execute NTA Blocks', monitors student actions with regard to the training objectives selected for the training exercise. Navy Training Task (NTA) blocks are used to represent these training objectives within our models. Each NTA block encapsulates a particular training objective. For example, as shown in Figure 6, NTA block 2.1.1

monitors the student's ability to effectively use his available sensors upon initial detection of a subsurface contact in order to establish confidence level and classification.



NTA Number	NTA Name	Training Objective Description
NTA 1.2.1	Water Space Management	Tests how quickly students establish a No Attack Zone
NTA 2.1.1	Collect Target Information	Tests students' use of available sensors upon initial detection of a subsurface contact to establish confidence level and classification
NTA 5.1.1	Display Tactical Picture	Tests students' use of displayed tactical picture to maintain situation awareness
NTA 6.1.1	Protection of Naval Forces	Tests students' utilization of appropriate defensive formations, screens, and maneuvers

Figure 6. NTA Blocks in VO/C Model

The VO/C model assists the instructor by monitoring student performance and intervening in training as necessary, ensuring that the training session covers all of the training objectives encapsulated within the selected NTA blocks. In order to determine if intervention in training is necessary, the VO/C model monitors students for "trigger actions," actions that students exhibit which may indicate that they need help. When a trigger action is observed, the VO/C model reflects upon the student's cognitive process model to determine *why* the student performed the trigger action.

Before the training scenario begins, the instructor answers questions about the student's proficiency level on skills pertinent to the objectives being trained. Based on these responses, the VO/C selects the cognitive process model it determines will most closely represent the student's cognitive reasoning patterns during training. Human instructors recognize that observed student actions may be the result of many possible cognitive reasoning paths and are able to provide student-specific guidance accordingly. By selecting the cognitive process model that matches the student's, the VO/C model is able to provide guidance tailored to the student's ability level much in the same way a human instructor would. Two example cognitive process models representing the reasoning involved for the "Collect Target Information" objective are shown in Figure 7.

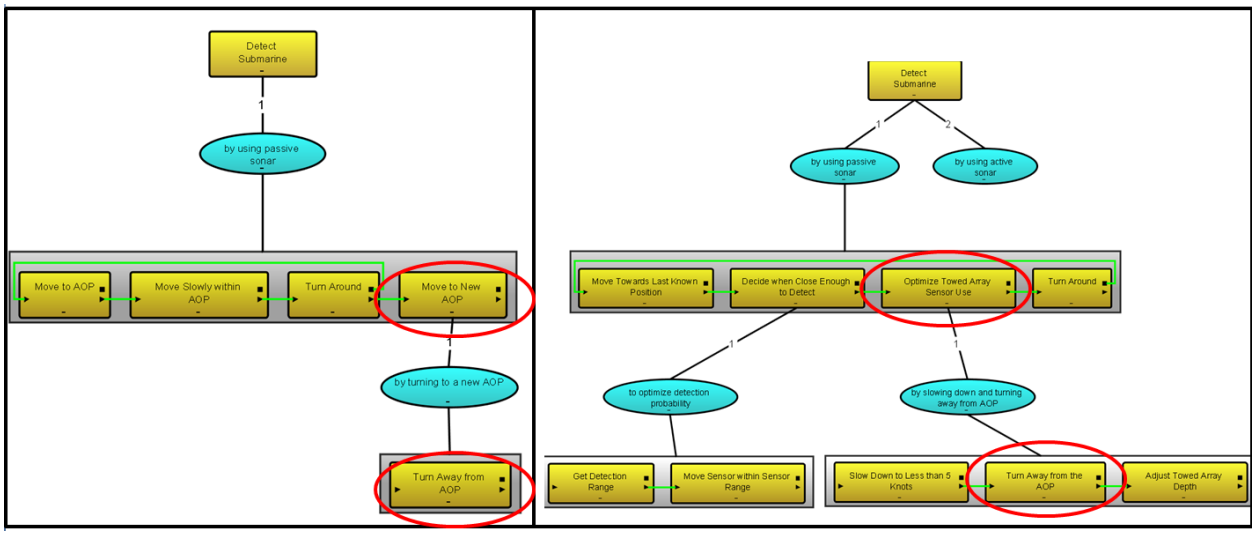


Figure 7. Novice Students vs. Advanced Students' Cognitive Process Models

Figure 7 shows the student cognitive process models with the task containing the trigger action of "Turn Away from the AOP (Area of Probability)" and parent task circled in both models. The reason the VO/C model would infer for this trigger action would depend on the cognitive process model selected for the student. If the novice cognitive process model were selected, shown on the left in Figure 7, the VO/C model would infer the reason the student turned his ship to be to move to a new Area of Probability (AOP) and the VO/C would intervene in training. Conversely, if the higher proficiency cognitive process model were selected, shown on the right in Figure 7, the VO/C would infer the student was turning away from the AOP to optimize use of a towed array and the VO/C would not intervene.

In addition to intervening when a student performs a trigger action, the VO/C alerts the instructor and intervenes during training based on general scenario information. For example, if the student is in the process of collecting target information, half the training scenario time has elapsed and the student has yet to make contact on its target, the VO/C model will adapt the scenario to make it more likely for the student to make contact on his target.

VO/C models intervene and adapt training scenarios by communicating with our behavior model-driven entities in the synthetic training scenario to alter their behavior. For example, if the VO/C observes that the student is moving to the wrong AOP and there is a limited amount of time left to locate the enemy submarine in the scenario, the VO/C may communicate with the submarine, instructing it to change its course to move closer to the student to make it easier to detect. A framework for sending these communications is described later in this paper.

Simulation Integration

We have been able to deploy our behavior models and VO/C models to drive the behavior of entities within a variety of simulations, including JSAF and Virtual Battle Space™ (VBS). This integration is accomplished through the use of a simulation interface layer which provides the following core functionality:
1. Action functions- functions called by behavior model and VO/C model procedures to cause the entity to perform actions in the simulated environment, such as moving to a particular location, turning sensors on/off, or firing a weapon.
2. Query functions- functions called by behavior model and VO/C model procedures to request information about the simulation, such as the current state of the entity driven by the behavior model or its sensor stimuli.

A differentiator between action functions and query functions is that query functions expect a response from the simulation while action functions cause action within the simulation. The key point to note is that the behavior models are simulation agnostic and can be adapted to integrate with various simulation environments. If multiple simulations support the same set of sensory input and actions required by a particular behavior model, that model can be deployed in both simulations without modification. Additionally, the TMK structure of the behaviors

supports failure based decision making so that a given task can have multiple methods of completion, each attempted in series until one succeeds in achieving the goal of the task. Thus, a given task may specify less efficient methods of completing its goals, which the system can call back upon if the integration with a particular simulation environment does not support all of the functionality required by the preferred method (Potts *et. al.*, 2010).

Communication Framework

Our communication framework enables behavior model-driven entities in the training scenario to handle communication from human users as well as other autonomous entities by extending the notion of BLAs. On previous projects, we have developed communication handlers, represented as data objects attached to individual BLAs. The decision to tie an entity's communication capabilities to the BLAs driving its manifested behavior conferred a considerable advantage: our behavior models are able to contextually separate various kinds of communications. Instead of forcing a behavior model to consider all possible communications which could take place within any possible scenario, the behavior model can focus on communications which are contextually specific to a particular type of action.

The communication handler consists of two primary components: a TMK hierarchy representing the process required to handle incoming communications and a set of data representing information important to the types of communication which are relevant within the context in which the entity is performing the BLA.

During development of individual BLAs, a corresponding communication handler is also created, and performs the function of defining how the autonomous entity will communicate with other entities, both human and autonomous, while in the process of performing that BLA. For example, while performing a BLA designed to move an entity to a particular location, the BLA's communication handler may contain information about the entity's current destination, and be able to answer questions about it. While performing a different BLA, such as firing on a ground target, a different communication handler would be in effect, no longer containing information about a destination (which would have no meaning in its current context), but instead containing information about its current target, the weapons it was deploying, etc. (Potts *et. al.*, 2012).

VIRTUAL OBSERVER CONTROLLER FOR ADAPTIVE TRAINING (VOCAT) USE CASE

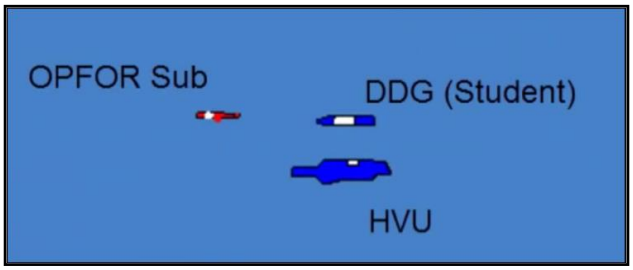


Figure 8. JSAF Training Scenario

We constructed a synthetic scenario, similar to training event scenarios we have observed at the Fleet ASW Training Center, San Diego, with the purpose of testing the technical merit of our VO/C framework. We began by using the authoring console to construct a VO/C model, called the Virtual Observer Controller for Adaptive Training (VOCAT), containing NTA block 2.1.1, "Collect Target Information." Our synthetic scenario, constructed in JSAF, contained a student ship, an enemy submarine, and a high value unit (HVU), as shown in Figure 8. We assigned the submarine the JSAF behavior of moving on a heading away from the student. We manually controlled the student ship and had the HVU moving with the student. We played the role of the instructor, entering input into the student and scenario information screens. We selected novice inputs for each of the student input categories and ran the scenario for ten minutes for demonstration purposes (a real training event would be on the order of 60 minutes).

After 50% of the scenario run time had elapsed, we (playing the role of the student) had yet to localize the submarine. We observed that VOCAT had updated the instructor with the action it was performing in order to help the student, which was having the submarine ping to help the student find the submarine. After 70% of the scenario run time had elapsed, we observed that the instructor had been alerted that the student had not yet found the submarine. VOCAT had also told the submarine to fire a torpedo to help the student find it, as shown in Figure 9.

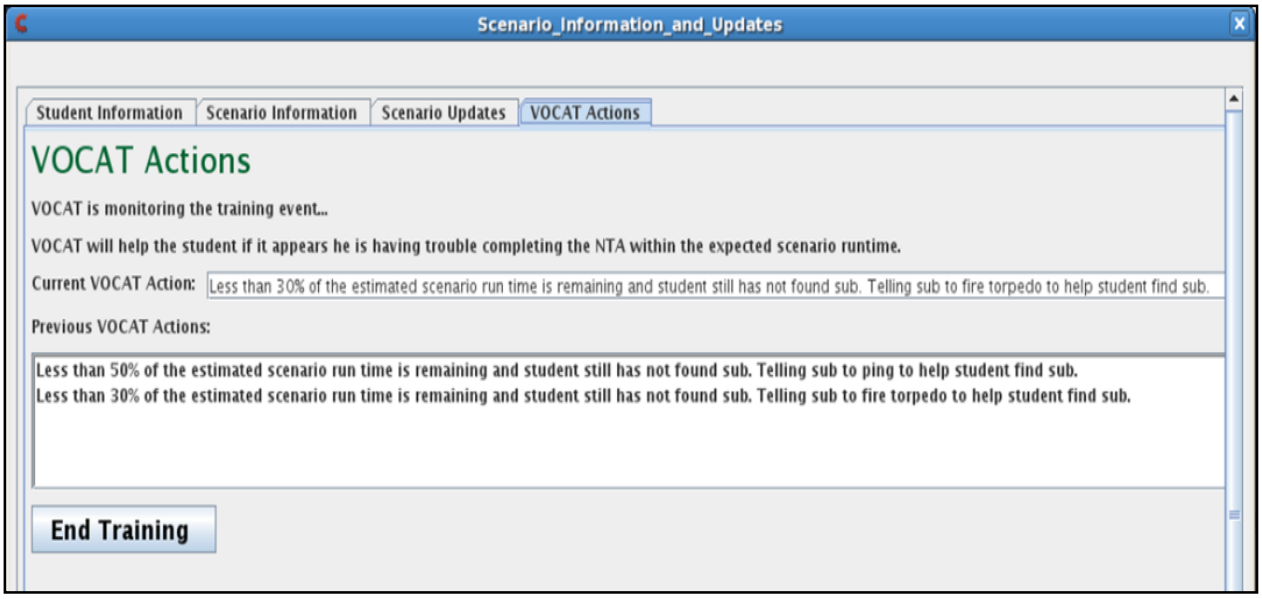


Figure 9. VOCAT Student Actions

In the remaining few minutes of the scenario, we turned the student ship away from the submarine. As a result, VOCAT successfully alerted the instructor to this action and displayed possible reasons to explain this action, as shown in Figure 10. Upon reflecting on the cognitive process model selected for the student and determining that the student needed help, VOCAT turned the submarine in the direction of the student to further help the student.

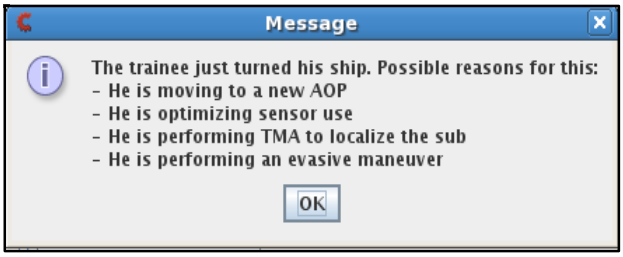


Figure 10. Alert Message to Instructor

After 100% of the scenario had elapsed, VOCAT ended the training scenario and created a file containing key information from the training scenario. This report contained accurate distance measurements from the student ship to the sub throughout the training scenario, which is information that SMEs have identified as key to ASWE instructors. VOCAT also documented the closest distance the student got to the sub along with the time at which it occurred. In addition, it documented the trigger action performed by the student, the reason for the action, and the time at which it occurred.

DISCUSSION

Our ASW synthetic training event use case demonstrates that our underlying framework supports our VO/Cs running end-to-end in synthetic training scenarios. In addition, our use case demonstrates that our VO/Cs perform the five capabilities of effective VO/Cs described in the introduction of this paper. First, our VO/Cs monitor student performance with regard to their training objectives, as specified by the NTA blocks used to build the V/OC model. Our VO/Cs also alert the instructor to key scenario and student actions in real time, as shown in Figure 9 and Figure 10 and described in the use case. Second, because students can perform a variety of actions during training for different reasons, our VO/Cs reflect on student cognitive process models in order to understand why students are performing certain actions. Third, our VO/Cs determine how and when to intervene during training depending on the reasoning behind students' actions, freeing the instructor from having to do so. Fourth, our VO/Cs effectively intervene in the training scenario based on students' proficiency levels and the tasks to be completed. As described in our use case, the VO/C intervened in the scenario in order to make the enemy submarine's location more obvious to the student, first by sending a message to the submarine to ping, second by sending a message to the submarine to fire a torpedo, and third by sending a message to the submarine to move toward the student. Fifth, our VO/Cs save

key information from training sessions, including relevant entity distances, trigger actions, and the inferred reasons behind students' actions, to aid the instructor during AAR.

CONCLUSION

In this paper, we have presented a framework for developing adaptive, intelligent VO/Cs built to aid instructors in helping naval students meet training objectives during synthetic training. Through use of the behavior authoring console and BLAs, instructors can create behavior models to drive the behaviors of automated entities used in synthetic training. Similar to behavior models, VO/C models are authored using NTA training blocks the instructor selects to cover during training. The VO/C model assists the instructor by monitoring student performance and intervening in training as necessary, ensuring that the training session covers all of the training objectives encapsulated within the selected NTA blocks. This intervention is achieved by adapting training scenarios through communicating, via communication handlers, with the behavior model-driven entities in the synthetic training scenario to alter their behavior. We have been able to deploy our behavior models and VO/C models to drive the behavior of entities within the JSAF and VBS simulations. Our technical feasibility test has demonstrated that our VO/C model, VOCAT, is able to run end-to-end and has the potential to be of aid to instructors.

We have visited Navy training facilities and witnessed several live training events. The next phase of our work is to expand our VO/Cs' capability to run in real training scenarios. This will entail creating a larger set of NTA blocks and student cognitive process models for our VO/Cs to use. Our work has also sets grounds for future work in the areas of adaptive, intelligent training and AAR. In particular, we are interested to see whether our VO/Cs use as a training aid will reduce instructor-to-instructor variation in training, and consequently, student learning. In addition, we are interested in allowing instructors to author their own NTA blocks and use their own expertise to determine how our VO/Cs should intervene during training. Finally, while we have focused our work on Navy ASW training, we are eager to expand our efforts to enhance training across the Defense sector, as the framework we have described in this paper is not domain specific. We look forward to working with others in the training and simulation community to advance these areas.

ACKNOWLEDGEMENTS

We would like to thank ONR and NAVAIR-TSD for their support of the work described in this paper.

REFERENCES

Endsley, M.R. (1995). Toward a theory of situation awareness in dynamic systems. *Human Factors, 37(1),* 32-64.

Porayska-Pomsta, K., Mavrikis, M., & Pain, H. (2008). Diagnosing and acting on student affect: the tutor's perspective. *User Modeling and User-Adapted Interaction, 18(1-2),* 125-173.

Potts, J.R., Griffith, T., Sharp, J.J., & Allison, D. (2010). Subject matter expert-driven behavior modeling within simulation. *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference.*

Potts, J. R., Griffith, T., Roth, K. , & Snyder, J. (2012). Customizable speech centers for automated entities within simulation. *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference.*

Rosch, E.H., Mervis, C.B., Gray, W.D., Johnson, D.M., Boyes-Braem, P. (1976). Basic objects in natural categories. *Cognitive Psychology, 8 (3),* 382-439.