Two Methodologies to Assess UrbanSim Scenarios

Brian Vogt	Joseph Sullivan, Ph.D.	Jonathan Alt, Ph.D.
Lieutenant Colonel, U.S. Army	Commander, U.S. Navy	Lieutenant Colonel, U.S. Army
Yorktown, VA	Monterey, CA	Monterey, CA
brian.vogt@us.army.mil	sullivan@nps.edu	jkalt@nps.edu

ABSTRACT

Turn-based strategy games and simulations are vital tools for military education, training, and readiness. In an era of increasingly constrained resources and expanding demand for training solutions, the need for validated, effective solutions will increase. Appropriate performance feedback is an important component of any training solution. Current methods for designing and testing the performance feedback provided in turn-based simulation are limited to well-structured problems and do not adequately address ill-structured problems that better replicate problems facing military leaders in today's complex operating environment. This paper develops and explores two new methods for assessing the feedback mechanisms of turn-based strategy games. Using UrbanSim, a game for training strategic approaches to counterinsurgency operations as an exemplar, this research developed and explored two unique methods for evaluating the reward structure of the UrbanSim scenarios. The first method evaluates different student strategies using a batch-run method. The second method uses a reinforcement-learning algorithm to explore the decision space. These scenario evaluation methodologies are shown to be able to provide insights about a game's performance feedback mechanism that was not previously available. These methodologies can be used for formative evaluation during game scenario development. Additionally, these evaluation methodologies are generalizable to other training and education games that focus on ill-structured problems and decision-making at discrete intervals.

ABOUT THE AUTHOR

Brian Vogt was commissioned an Armor Officer in 1996. He served as an armor officer in several leadership positions at Ft. Hood, TX, Ft. Knox, KY, Camp Casey, Korea, and Ft. Riley, KS. He commanded a tank company and head-quarters company in two separate tours in Baghdad. He served as a simulations operations officer since 2006 at Ft. Leavenworth, KS as a simulations analyst for the SE Core program. He is a graduate of the Armor Officer Basic Course and Advanced Course, Combined Arms Services Staff School, Command and General Staff College, and the Naval Postgraduate School where he earned a MS in Modeling, Virtual Environments, and Simulations. He is currently serving at Ft Eustis, VA.

Joseph Sullivan graduated from Catholic University of America with a Bachelor of Science in Computer Science, attended Aviation Officer Candidate School, and was commissioned in 1986. He has a Ph.D. in Modeling and Simulation from the Naval Postgraduate School (2010). CDR Sullivan's operational background includes multiple tours in east and west coast Helicopter Antisubmarine (HS) squadrons. In 1998, CDR Sullivan earned a Masters of Science in Computer Science degree at the Naval Postgraduate School. CDR Sullivan has been assigned to the Naval Postgraduate School since 2001. While assigned to NPS he has filled various roles including Military Faculty, Program Officer and Principal Investigator. His research background and interests center on the application of emerging virtual environment technology to training.

Jonathan Alt is an operations research analyst currently assigned as a military instructor at the Naval Postgraduate School. LTC Alt served as an infantry officer in a variety of positions at Fort Stewart, Georgia and Fort Campbell, Kentucky. As an operations research analyst LTC Alt most recently served as the director of TRADOC Analysis Center, Monterey.

Two Methodologies to Assess UrbanSim Scenarios

Brian Vogt	Joseph Sullivan, Ph.D.	Jonathan Alt, Ph.D.
Lieutenant Colonel, U.S. Army	Commander, U.S. Navy	Lieutenant Colonel, U.S. Army
Yorktown, VA	Monterey, CA	Monterey, CA
brian.vogt@us.army.mil	sullivan@nps.edu	jkalt@nps.edu

INTRODUCTION

The use of games and gaming to educate is certainly not new. Games have been used in educational settings for many years with varying levels of success. Many times these games have focused on well-defined problems such as math, science, and procedural trainers. The reward structure of these types of games can be directly validated if they reward the student with the one correct answer or solution. However, there has been an increased desire to use games to train and educate students to perform well in ill-defined problem areas. Ill-defined problems are characterized as having more than one correct, or acceptable, solution. Validation of games that address ill-defined problems is inherently more difficult than well-defined problems. One of the challenges in the application of complex agent based games built for training and education is the verification that the intended learning outcomes are being reinforced by the training system, and likewise that undesired behaviors are not being rewarded. This paper will compare two methodologies to address this challenge: one that looks at the descriptive statistics of the end of run metrics from a sample generated from the same strategy (a batch run) and another that employs reinforcement learning to fully explore the decision space.

The U.S. Army’s use of a game called UrbanSim provides an example of a use case involving a complex agent based model. UrbanSim is a turn-based strategy game that is designed to train leaders in executing battle command in complex environments focused on counterinsurgency and stability operations (Wansbury, 2011). UrbanSim was developed and fielded by the U.S. Army as a tool to support educational objectives concerning counterinsurgency operations at the School of Command Preparation at Fort Leavenworth, Kansas. A reasonable method to evaluate the scenarios and the performance feedback mechanisms was not readily available to the development team (Wansbury, 2011).

There is limited direct evidence to support that the scenarios developed and fielded supported the educational objectives. That is to say, that the embedded performance feedback mechanisms within UrbanSim has not been evaluated to ensure students were guided through rewards and penalties to achieving better understanding of counterinsurgency operations. The development team assumed risk in this area because UrbanSim was intended to be used in the classroom with an instructor. If the results of actions in the game did not seem correct, or falsely rewarded poor decisions, the instructor was able to give verbal feedback to overcome this apparent shortcoming of the UrbanSim scenario performance feedback. Additionally, scenario validation did not seem feasible at the time of fielding due to the vast number of possible ways to play the game. The use of UrbanSim has grown from a simulation to support Fort Leavenworth’s School of Command Preparation under the supervision of an experienced instructor to being used at Captain Career Courses, Non-Commissioned Officer Academies, Service Academies, as well as available to all Soldiers via the Army Military Gaming website. These expanded uses reduce the role of an experienced instructor that can guide students when the results of the game are contrary to desired learning objectives. The increased use of UrbanSim and other games for training as unsupervised learning platforms highlights the need to ensure the performance feedback mechanisms in training and educational games properly reward good performance and penalize poor student performance.

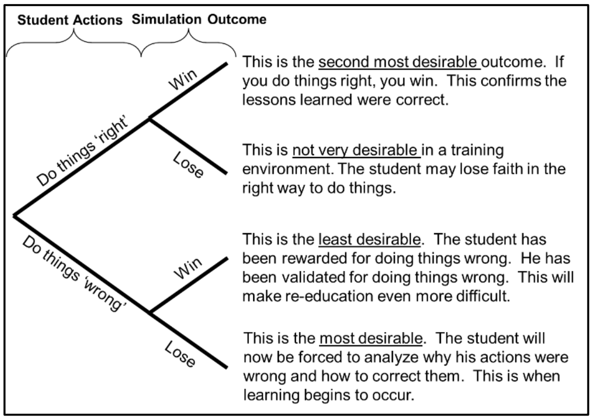
UrbanSim is a good test-case of a larger problem with simulations and games for education. UrbanSim was designed for use with an instructor guiding the learning experience. However, UrbanSim is now fielded and available without instructors. If we can figure out what is missing or needed to effectively use UrbanSim without instructors, we will make progress toward designing effective simulations and games for use without instructors.

BACKGROUND

James Ong stated that “Practice and experience, whether simulated or on the job, are not enough to ensure effective learning. Learners must be able to make sense of those experiences to identify poor decisions and actions, missing knowledge, and weak skills that deserve attention” (Ong, 2007).

Perhaps the most critical component of Ericsson’s deliberate practice model is performance feedback (Ericsson, 2008). Performance feedback encompasses more than just a message that you completed the exercise successfully. Performance feedback includes everything the learner perceives that helps them make connections between their actions (cause) and the outcome of those actions (effect).

There are many ways to provide performance feedback to the student during and after an exercise to influence learning. For well-defined problems, the tree diagram in Figure 1 describes the notion that games, as well as all training and education, should reward good performance and penalize poor performance. Additionally, there are negative consequences to rewarding poor performance and penalizing good performance.



This is the second most desirable outcome. If you do things right, you win. This confirms the lessons learned were correct.

This is not very desirable in a training environment. The student may lose faith in the right way to do things.

This is the least desirable. The student has been rewarded for doing things wrong. He has been validated for doing things wrong. This will make re-education even more difficult.

This is the most desirable. The student will now be forced to analyze why his actions were wrong and how to correct them. This is when learning begins to occur.

		Performance Feedback	
		Reward	Penalty
Student Performance	Correct	Desirable	Not Desirable – student received negative reinforcement feedback from correct performance
	Incorrect	Not Desirable – student received positive reinforcement feedback from incorrect performance	Desirable

Figure 1. Performance Feedback Tree Diagram and Matrix for Well-Defined Problems

This tree diagram can also be represented in a matrix that is analogous to statistical Type I and Type II errors. Type I error can be thought of as providing negative feedback for correct performance, and Type II error is represented by providing positive feedback for incorrect performance.

Performance feedback for ill-defined problems is not as straight forward as it is for well-defined problems. Clark describes ill-defined tasks and problems as “scenarios or cases for which there is no one correct answer or approach… ill-structured problems are considered best for problem based learning” (U.S. Army, 2011). Ill-defined problems are also characterized as problems where there exists a range of acceptable solutions and a range of unacceptable solutions. In the range of acceptable solutions, the solutions may be very different from each other, but still adequately address the problem and should be rewarded equally.

Figure 2 graphically depicts this notion as it relates to performance feedback. The “unacceptable performance” region of this curve refers to performance that is unacceptable and is used to identify students that do not have a requisite knowledge to begin deliberate practice. The learning portion of the curve is very important for student learning. This region is where students depend on the reward associated with their performance to gain insights about which strategy is better than other strategies. The acceptable performance region indicates where student performance matches the desired training or educational goals of the exercise. This curve is utilized, in practice, in the entertainment game industry to keep players in what Murphy refers to as “flow” or the learning portion of the curve (Murphy, 2011). This supports the intrinsic rewards found in play by Ericsson (Ericsson, 2008).

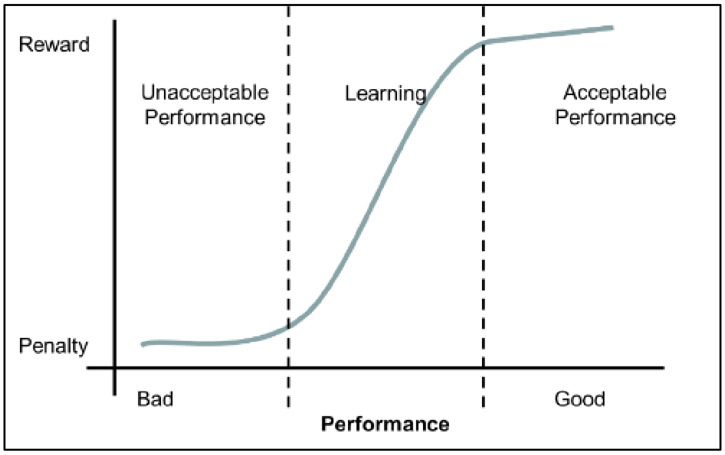


Figure 2. Reward Function as it Relates to Performance

The reward function curves can also be used to evaluate existing training simulations and scenarios. The following charts show a few hypothetical reward functions that do not support the desired training objectives.

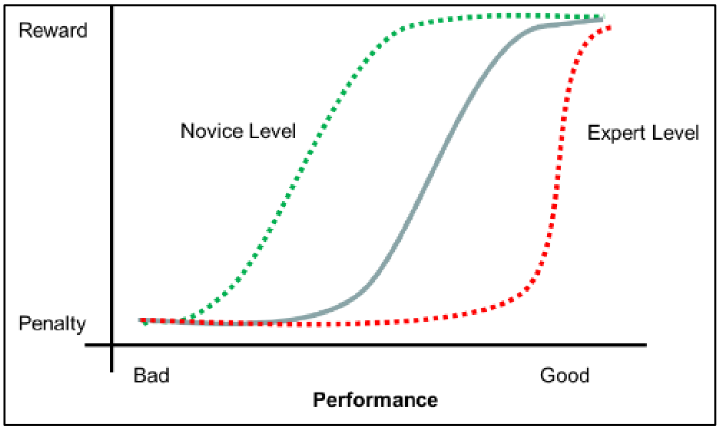


Figure 3. Manipulation of Reward Curve for Games

Figure 4 describes a reward function that rewards mediocre performance over good performance. This is undesirable because students would perceive their mediocre performance as the desired good performance.

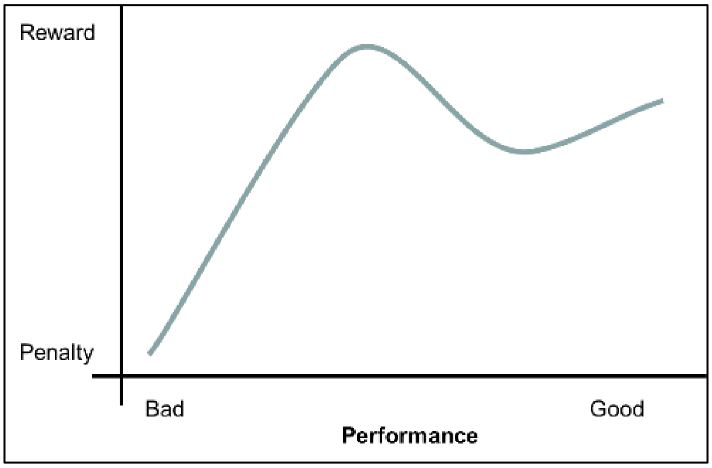


Figure 4. Undesirable Reward Function That Rewards Mediocre Performance

Figure 5 describes a reward function that does not adequately differentiate good performance from bad performance. This is undesirable because students perceive that there is no way to "win" and no way to "lose" so they do not adjust or improve their performance to obtain good performance.

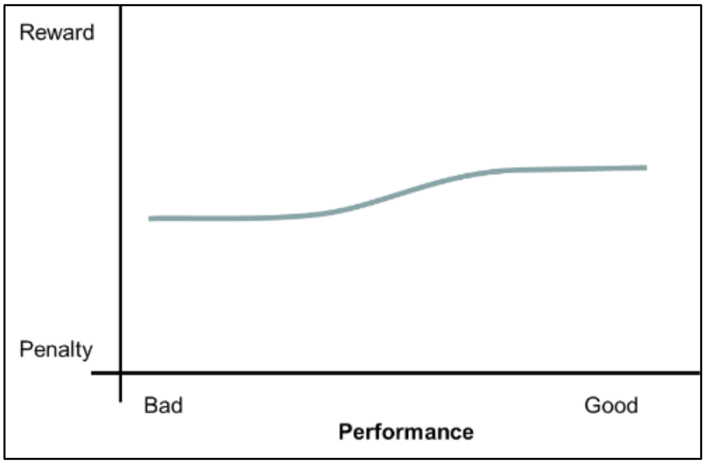


Figure 5. Undesirable Reward Function That Does Not Adequately Differentiate Between Bad and Good Performance

EXPERIMENTAL METHODOLOGY

Methodology to evaluate games and scenarios that address ill-structured problems

The following methodology was developed to evaluate UrbanSim scenarios for this research effort. However, this general methodology could be used, or adapted, to evaluate other games and scenarios.

1. **Identify the training objectives**. The training objectives are usually described in terms of what performance the learner should perceive as a reward. However, it is equally important to understand what performance the learner should perceive as a penalty.

2. **Identify the possible learner strategies**. This should span all of the possible ways of playing the game to ensure a more complete understanding of the reward signal. However, there may be times when only a small subset of strategies is appropriate to analyze. In general, all possible strategies should be explored when the intended learner is a novice. Whereas, the training developer may limit the scope for analysis if the intended learner is an expert and will focus their decisions on a smaller decision space. Additionally, if the training objectives call for a specific action to take place at a specific time or event in the scenario, this can also be evaluated.

3. **Identify which of the possible learner strategies should be rewarded and which strategies should be penalized**. This does not have to be precise at this point, but can assist with identifying what possible learner strategies should be evaluated. This analysis should explicitly reflect the training objectives.

4. **Develop the means to batch run the games with an automated tool**. This may result in considerable amount of work if it is not created already. Ideally, the game should be able to run automatically from the command line.

5. **Run the game and collect the data.** The data collected should identify the strategy or policy used and the result. The result may be a score, a quantifiable outcome, and any other means of quantifying performance. The result used should mirror the result that the learner will see as a part of the game's performance feedback mechanism. Using the brute force method, a minimum of 30 runs of each strategy is desirable to use the central limit theorem as a part of the analysis.

6. **Analyze the data.** Use a statistical analysis software package to understand the mean and standard error of each strategy. Organize the results in rank order. Then compare the different strategies to each other. Look at the list of

strategies and determine if 1) only acceptable strategies are among the highest rewarded strategies and 2) only unacceptable strategies are among the least rewarded strategies. This ensures that good performance is rewarded and poor performance is penalized.

7. **Adjust the scenario or reward function of the game or scenario as needed**. If bad performance is inadvertently rewarded or good performance is penalized, there is a problem with the scenario or game that produces this result. The scenario designer must redo the experimental runs after any changes are made to the scenario or game to ensure no inadvertent mistakes were made during the editing.

Technical Approach

The UrbanSim game is composed of the graphical user interface that is unique to UrbanSim. Within the UrbanSim game, PsychSim is the simulation model that is used to adjudicate the user actions and impact on the game environment. Python code from David Pynadath, was modified to interface with the UrbanSim's PsychSim software to conduct the experiments. This code enabled the simulation experiments to run from the command line, which in turn enabled batch running as well as reducing the time to play the game from roughly an hour per game to approximately one minute. Figure 6 depicts the existing UrbanSim practice environment and the software components added to execute the experiments. Shaded area depicts the existing practice environment. The components outside the shaded area were added to conduct the experiment.

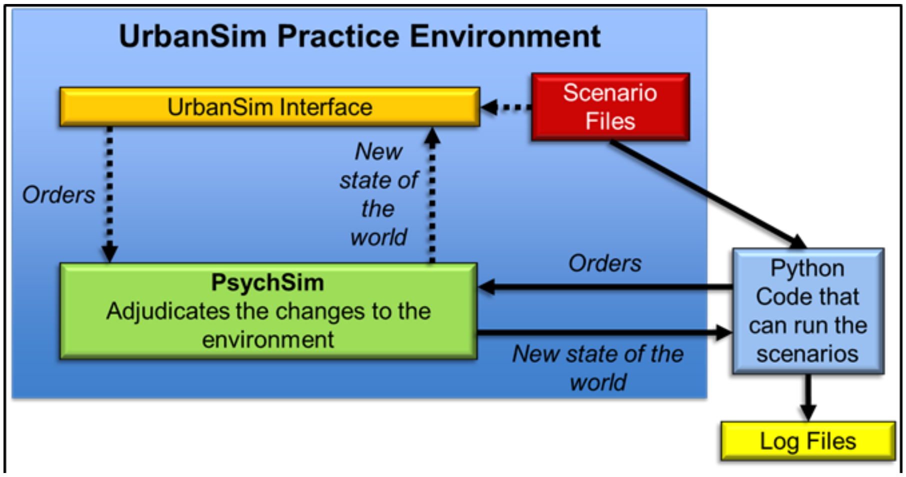

UrbanSim Practice Environment
UrbanSim Interface
Scenario Files
Orders
New state of the world
PsychSim
Adjudicates the changes to the environment
Orders
New state of the world
Python Code that can run the scenarios
Log Files

Figure 6. The Experiment Configuration.

Batch Run Method Experiment Design

The most extensive experiment, using a full factorial design of experiments with five factors, was examined a five digit strategy that explored 162 different potential student strategies. Figure 7 portrays how the strategies were created based on the stated training objectives of the scenario. The first digit describes if the strategy was strictly legal actions versus a mixed legal/illegal actions. The second digit describes if the actions were strictly lethal, nonlethal, or a mixture of lethal and nonlethal. The third through fifth digit describe the type of actions of the first, middle and last third of the game. For example, a 'srchb' strategy means that it the strategy is exclusively legal (s), it is a mixture of lethal and nonlethal actions (m), first five turns are 'clear' type actions (c), middle five turns are 'hold' type actions (h), and last five turns are 'build' type actions.

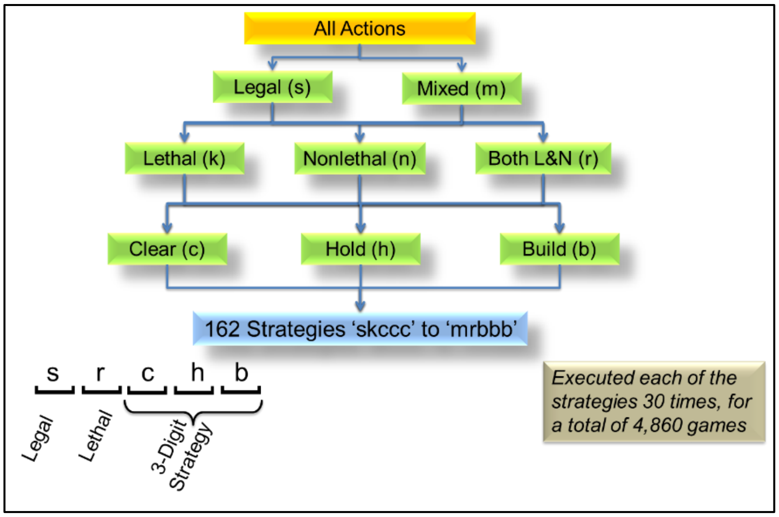


Figure 7. Five-digit strategy development

The experiment consisted of 30 replications for each of the 162 different strategies, for a total of 4,860 games. The final score of each game was stored in a data file for analysis.

Reinforcement Learning Method Experiment Design

This experiment used the same 162 different strategies that were used in the 5-digit batch experiment. However, instead of running 30 replications of each strategy, a reinforcement-learning algorithm explored and gained insight about the underlying reward structure. The experiment used an epsilon-greedy exploration policy. The epsilon-greedy policy selects the strategy with the best observed score, from the 162 different strategies, with a proportion of 1-ϵ of the number of trials. The value for ϵ was 0.1, meaning that 10% of the time the agent will take a randomly selected strategy, and 90% of the time the agent will select the highest valued strategy. The experiment used the Direct-Q Computation (DQ-C) method for the value function. The reward function was at the end of the 15-turn game.

The experiment ran for 10,000 iterations with the first 5,000 iterations using a randomly selected policy. The last 5,000 iterations used an increasingly greedy strategy selection, meaning that the value of ϵ decayed over time. The key data collected from this experiment is the value estimates of the strategies. The value estimate of the strategy is the discounted average of the scores of the previous games using the particular strategy. The value estimate is not the expected score of the strategy (Alt, 2012).

RESULTS

Batch Run Method Experiment Results

The data from the batch run method was analyzed using a JMP 9.0 Pro, which is a statistical analysis software package. Each of the scenario training objectives was evaluated based on the statistical analysis of the game results. Figure 8 is one example of the analysis from the batch run method. Scenario designers can use this output to ensure that only the most desired strategies are rewarded and only the least desired strategies are penalized. The scenario designer will need to ensure that the resultant reward function curves resemble Figure 2, and not Figures 4 or 5. In this case, the most desired strategy of 'clear, hold, build' (chb) is the ninth best rewarded strategy and 'build, build, build'(bbb) strategy is the best rewarded strategy, therefore it resembles Figure 4 where mediocre performance is rewarded better than the desired performance.

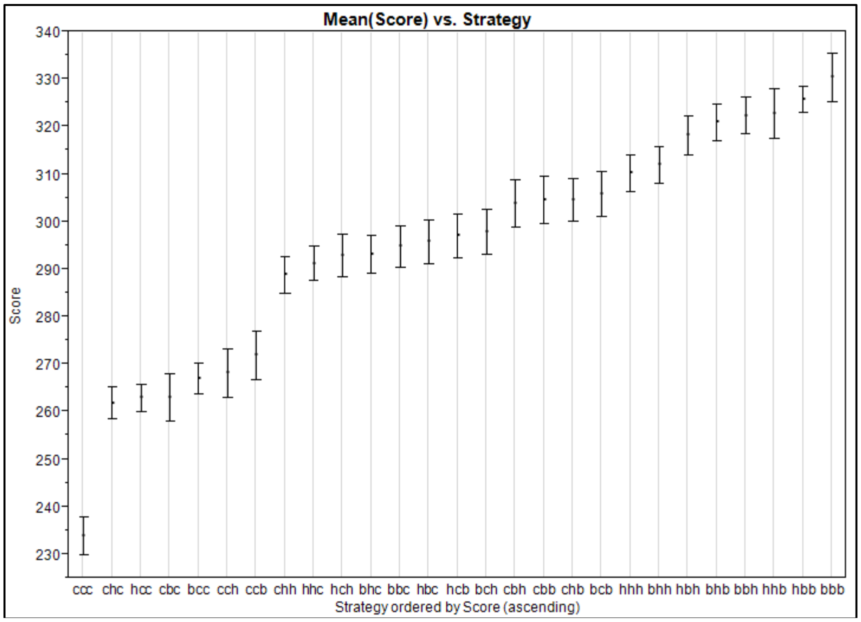


Figure 8. Plot of the mean score versus strategy with standard error bars.

Reinforcement Learning Method Experiment Results

The reinforcement learning method produced the following plots for the scenario designer to use. The histogram plots determine the frequency the various strategies were used by the reinforcement learning algorithm. The greater the frequency indicates the greater perceived value of the strategy. When the reinforcement learning algorithm quickly eliminates most strategies, and settles on a few strategies, the scenario has a strong signal to noise ratio. That indicates that the novice learner should be able to quickly discern appropriate strategies from inappropriate strategies in fewer game iterations. This may not reflect reality outside of the game environment, but would aid in achieving learning objectives. Conversely, a low signal to noise ratio would challenge the more experienced player to pick out the reward signal in a greater amount of noise. Therefore, referring back to Figure 3, we can use this information to create scenarios that keep players engaged in learning.

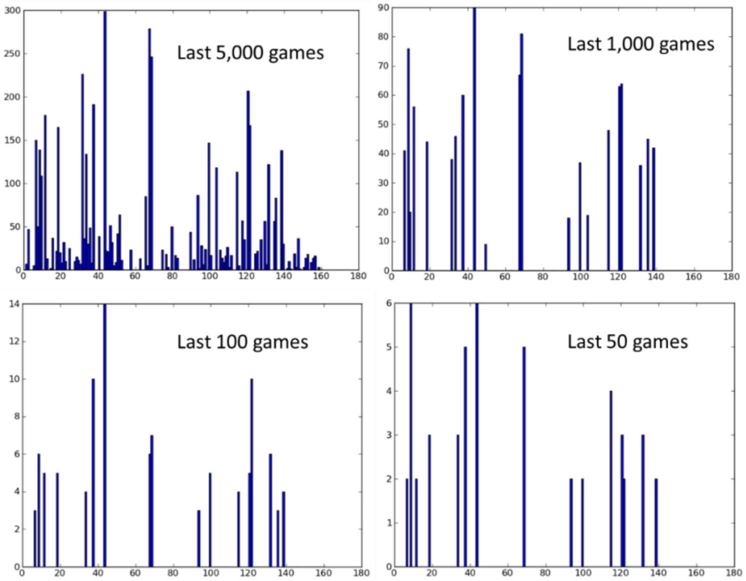


Figure 9. Histograms of the reinforcement learning algorithms. The x-axis represents the strategy index, and the y-axis is the frequency the strategy was valued the greatest.

From the perspective of evaluating the fielded UrbanSim scenarios, it appears that the unstated, but assumed, training objective of rewarding students that conduct exclusively legal actions is properly rewarded. The training objective of emphasizing the doctrinal principle of “Clear, Hold, Build” did not stand out very clearly. However, it appeared to be in the range of acceptable solutions. The fact that the Build, Build, Build strategy was also in the range of acceptable solutions is not desirable because it reinforces the notion that you can be successful if you ignore the enemy and allow them to operate and you can still be successful in the scenario. The fourth training objective wants the students to demonstrate that a mixture of lethal and non-lethal actions is better than exclusively lethal or exclusively non-lethal was not supported. Non-lethal actions were more strongly rewarded than the mixed approach and the lethal actions. This may be closely tied to the fact that the enemy units in the scenario do not affect the simulated environment enough to replicate the danger of ignoring enemy units operating in the area of operation.

The approach of using automated tools to evaluate a game or game scenario provides insight to the scenario author. Additionally, evaluating a scenario with respect to the training objectives is a necessary step with all training games, but especially true of games that address ill-defined problems. The traditional approach of evaluating scenarios was to define and articulate training objectives, then develop the training scenario, make sure it functions, then use humans to play the scenario, and evaluate the game or scenario based on the training transfer that occurred within the participants. This process is rather resource intensive and can take a considerable amount of time. This approach of using automated tools to evaluate scenarios seeks to reduce the resources and time needed to evaluate training scenarios.

CONCLUSIONS AND FUTURE WORK

In general, both evaluation methodologies are able to provide insights about the performance feedback mechanisms in training scenarios that were not available before. The methodologies can assist scenario authors throughout the scenario design effort. Similar in nature to the computer programming axiom of “build a little, test a little,” this methodology allows scenario authors to conduct formative, automated testing to ensure the performance feedback mechanism supports the desired training objectives. This methodology provides a means of thoroughly testing and tuning a scenario before human participants begin play testing.

There are advantages and disadvantages to these two methodologies that should be considered carefully by scenario designers. When the number of possible strategies is relatively small, in this case 162 strategies, the full factorial batch run is feasible to evaluate each strategy. However, when the decision space is very large, the reinforcement learning approach becomes more attractive. . The reinforcement learning methodology can be used to analyze the signal to noise ratio in the performance feedback signal, which in turn can be used to assess how long it could take for the learner to identify and apply the desired strategies.

These methodologies also have potential shortcomings as well. First, these methodologies require an ability to bin all of the actions available to the learner. Second, games that are not discrete time steps also present a challenge to this methodology. UrbanSim has 15 discrete turns for the player to make decisions. While the player is making decisions the environment is static and does not continue to change. Game scenarios that continuously change with time, such as driving or ship handling games, create a new challenge for the learner. Thus, this presents a new dynamic for the scenario designer to consider during design and testing.

In a different application, this methodology could be applied to evaluating training and education scenarios that address major combat operations. This was the original endeavor of this study, however, it seemed that the decision space was far too large and a game with 15 discrete turns was more manageable. As discussed earlier, the decision space within UrbanSim is deceptively large. Eleven units with between 140 and 341 possible actions over 15 turns generate more than $5x10^{27}$ possible ways of playing the game. In retrospect, a major combat operations game scenario may be easier to evaluate and provide performance feedback. For a division level scenario there may be 20–25 battalion sized units or units directly controlled by the division which is more than the number of units in UrbanSim. There also may be a few more decision points in the game when the player would give orders. However, for each unit there would be significantly fewer than 341 available actions for each unit, which would drive the decision space down to a manageable level. Using a similar approach of binning actions, the player could give orders to units like “move” to a pre-identified location, “attack” an enemy unit, “shoot indirect fire” at an enemy unit, etc., without having to get into the near infinite possibilities of where the unit is moving. Scoping this decision space would not negatively influence the student’s decisions, but would certainly make validating the scenario and providing feedback to the student more manageable.

ACKNOWLEDGEMENTS

This paper is based on the authors’ research work at the MOVES Institute at the Naval Postgraduate School. This effort could not have been done without Stacy Marsella and David Pynadath who provided expertise with UrbanSim and the underlying simulation, PsychSim.

REFERENCES

Alt, J. (2012). *Learning from noisy and delayed rewards; the value of reinforcement learning to defense modeling and simulation.* Monterey, CA.

Clark, R. C. (2008). *Building Expertise: Cognitive Methods for Training and Performance Improvement* (3rd ed.). San Francisco, CA: Pfeiffer.

Ericsson, K. A., Krampe, R. T., & Tesch-Romer, C. (1993). The role of deliberate practice in acquisition of expert performance. *Psychological Review*, 363–406

Fullerton, T. (2008). *Game Design Workshop: A Playcentric Approach to Innovative Games.* Oxford, UK: Morgan Kaufmann.

Intelligent Automation Incorporated. (2011). *Performance Assessment for Complex Simulations.* Orlando, FL: U.S. Army RDECOM-STTC.

Murphy, C. (2011). Why games work and the science of learning. *Interservice, Interagency Training, Simulations, and Education Conference*.

Ong, J. (2007, November/December). Automated performance assessment and feedback for free-play simulation-based training. *Performance Improvement*, pp. 24–31.

Ong, J., & Ramachandran, S. (2003). *Intelligent Tutoring Systems: Using AI to Improve Training Performance and ROI.* San Mateo, CA.

U.S. Army. (2011). *Army Doctrine Publication 3-0, Unified Land Ope*rations. Washington, DC: U.S. Army.

U.S. Army RDECOM. (2011). UrbanSim Training Package. Orlando, FL: U.S. Army RDECOM.

U.S. Army. (2011). *TRADOC PAM 525-8-2, The U.S. Army Learning Concept for 2015*. Fort Monroe, Virginia: U.S. Army.

USC Institute for Creative Technology. (2012). Retrieved February 16, 2012, from http://ict.usc.edu/projects/urbansim

Wang, N., Pynadath, D., Marsella, S., Cerri, S., Clancey, W., Papadourakis, G. et al. (2012). Toward automatic verification of multiagent systems for training simulations. In *Intelligent Tutoring Systems* (pp. 151–161). Berlin / Heidelburg: Springer.

Wansbury, T. (2011). UrbanSim Project Leader, U.S. Army Research, Development, and Engineering Command. (B. Vogt, Interviewer)

Wansbury, T., Hart, J., Gordon, A. S., & Wilkinson, J. (2010). UrbanSim: training adaptable leaders in the art of battle command. *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2010*, (pp. 1–10). Orlando, FL.