AI for Automated Combatants in a Training Application

Cindy Cox and Daniel Fu, Ph.D.
Stottler Henke Associates, Inc.
951 Mariner's Island Blvd., Suite #360
San Mateo, CA 94404, USA
+1 (650) 931-2700
[CCox,Fu]@stottlerhenke.com

ABSTRACT
We are trying to emulate the most realistic human-combat behaviors possible in a virtual environment. Our requirement for realism stems from our application: a combat training environment where the virtual arena needs to engage students in experiences that can carryover to "real life." For us, "realism" is focused on the behaviors of enemy combatants in the virtual world, and the definition of that reality needs to address questions like: Do the enemies hide from you? Where? How well? What weapons do they choose? How accurately do they aim? How fast can they shoot? Are they familiar with the tactics you and your team members use most often? Are they acting rationally given current conditions in terms of visibility, noise, and obstacles? In this paper we describe the system we are building: design choices, authoring concepts, and tools.

Categories and Subject Descriptors
I.2.6 [Artificial Intelligence]: Learning – *Concept learning*, *Knowledge acquisition*; I.2.6 [Artificial Intelligence]: Applications and Expert Systems – *Games*.

General Terms
Algorithms, Design, Experimentation, Measurement, Human Factors, Verification.

Keywords
NPC, Combat, Game Script, AI Engine, SimBionic.

1. INTRODUCTION
One of the greatest challenges in providing effective warfare training is the task of presenting realistic enemy forces. Full-scale war-gaming exercises will always be an essential component of combat training because of the unequaled benefits of practice against human opposing forces in a nearly-real battle situation. But with the increasing fidelity of virtual environments, computer training exercises can provide students with experience that may otherwise be too expensive, too difficult, or too dangerous to practice in the real world.

Realism is a common goal when creating virtual human characters, whether in a computer game, simulation, or educational tool, as the human users that interact with them have pre-existing expectations of how the characters *should* behave. We are trying to emulate the most realistic human-combat behaviors possible in a training application. Realism is a primary objective for us 1) to meet the expectations of users that computer characters behave in recognizably human ways and 2) to maximize the applicability and carryover of learned skills to actual combat.

Our automated characters battle against one or more users in an interactive, urban setting. The scenarios are geared towards Military Operations in Urban Terrain (MOUT) and the pitfalls of Close-Quarters Battle (CQB). In close-range engagements, many of the traditional strategies and weapons that have been long-utilized in extensive open areas are no longer effective, so CQB has unique strategies for success. Our automated Individual Combatants (auto-ICs) operate in the close quarters of multi-roomed city buildings. Users "clear" a building by examining all interior spaces in a prescribed tactical manner to ensure that enemy combatants do not remain in the structure.

Literature, observation, and Subject Matter Experts guide us in designing realistic behaviors and skills of opposing forces. Marine training manuals [1] recommend specific tactics to inspect and clear rooms of different sizes, hallway configurations, stairwells, and closets. We enable our virtual combatants to stress these strategies. In addition to the end goal of a cleared building, users get reinforcement training on documented techniques. Our application assesses visual and aural awareness of trainees, room breaching techniques, weapon aim, and reaction time, as well as the final defeat of enemies.

In the next section we describe our target application and virtual combat environment. Following that we discuss character design in terms of qualities auto-ICs can demonstrate, variables that affect their chosen actions during virtual battle engagements, and the synthesis of actions to compose complex behavior sequences. Next, we discuss technical design and implementation strategies. Finally, we define techniques and tools to validate scenario realism and tuning capabilities to customize our virtual combatants.

2. APPLICATION
Our combat technologies are collectively named "Shootwell." Shootwell is the deliverable of a contract for the Office of Naval Research's (ONR) Virtual Technologies and Environments (VIRTE) program. "VIRTE is an ONR program intended to research and develop a family of training simulators for Navy and Marine Corps expeditionary warfare… and demonstrate leap ahead human-immersive technology for naval training"[1] via the following goals:

[1] Darken, R. Office of Naval Research: VIRTE. Office of Naval Research, 22 September 2005: http://www.movesinstitute.org/darken/virte/index.html

- "Supplement & complement live simulations using virtual & wargaming simulations and other emerging technologies.
- Train warriors for ever increasing complexity and chaos.
- Significant time and cost savings.
- Rapidly transition superior technology to naval training systems for decisive military capability."[2]

Part of the VIRTE suite is a graphical application in the vein of a First-Person Shooter, designated the "Human Immersive Training Virtual Environment" (HIT VE), but commonly referred to by its executable program name, ManSIM. ManSIM is a multi-user, networked PC application that renders detailed exterior and interior terrains and can accept first-person control from a number of input devices such as head-mounted gear or weapon-shaped controllers as well as standard mouse and keyboard. It is through another networked simulation, JSAF, that Shootwell controls ManSIM's graphical characters or "avatars."

3. CHARACTER DESIGN

We want our human-like combatant characters in Shootwell to behave in ways expected by human users, but we are limited by how much of human cognition we actually are able to model. Development time and processing speed affect how much we truly *want* to implement as well. Actual combat includes the risk of being shot, which admittedly would be keenly realistic if included but also quite undesirable. With such limitations in mind, which parts of "humanness" matter most?

Our design includes specification of combatant behaviors that we believe contribute to the perception of reality in an urban combat setting. We have chosen to concentrate our characters' capacities on actions that reflect awareness of self and environment. The choices made programmatically for each character involve adaptation, planning, and variability. They are parameterized by unique characteristics of individuals in a given IC enemy population. Our goal is to construct observably unique and situation-appropriate maneuvers by the auto-ICs during virtual combat.

3.1 Awareness of Self and Surroundings

Simulated awareness guides Shootwell ICs in interacting with and reacting to stimuli in their environments. Some features that can make a character seem aware include:

- active exploitation of environmental features,
- minute response to stimuli,
- adaptation to user abilities and patterns,
- planning that is detectable in the execution of offensive or defensive maneuvers,
- and random behavior variability within a reasonable range.

[2] Darken, R. Office of Naval Research: VIRT E. Office of Naval Research, 22 September 2005: http://www.movesinstitute.org/darken/virte/overview.html

We assign unique identities to our virtual combatants in data profiles that contain personal attribute values. These profiles are stored in text files that can be readily created and modified. Some attributes initialize states and vary during program execution like hydration level, energy, injury, and duration of sustained combat. Other facets of human personality are statically defined for a character, such as intelligence and boldness. The characters' artificial intelligence analyzes their IC profiles to make decisions.

Characters can interact with their surroundings through access to a common store of data that is associated with their physical locales. The data includes area maps, obstacle markers, and the identification of inanimate objects for their usability as protection or improvised weapons. Using their pre-initialized data store, we try to make Shootwell ICs act as if they perceive their surroundings. Knowing that a wall is penetrable by the weapon it is carrying, an auto-IC might "decide" that firing through it is the right tactic to employ. Another possible behavior that shows a character's awareness is the use of inanimate objects. A decision might be made for it to move an item of furniture to barricade a room's entrance. Or, adversely, the virtual combatant might accidentally stumble over an object, creating noise that could be "overheard" by users.

To varying degrees, auto-ICs are motivated by self preservation. This is most apparent in their search for cover or concealment. Shootwell spatial-analysis algorithms identify locations in the vicinity of a character that are not visible from room apertures and are large enough to obscure the dimensions of the character's avatar. Location quality is ranked by bodily coverage, the protection afforded by blocking materials, and its proximity to exits. Finally, a location of a certain quality is assigned as the destination of the character seeking cover. The quality of the location assigned is dependent upon the skills and abilities of the virtual combatant and hence is not necessarily what has been computed to be the "best" location.

In terms of health, characters are affected both graphically and functionally even by non-fatal wounds. A shot in the leg can hobble a character without killing it. The character is then forced into a kneeling or prone position. Spreading affects of the injury to the character's anxiety level can affect both its action choices and the way actions are carried out.

3.2 Adaptation, Planning, and Variability

Unpredictability lends itself to variety and surprise. This not only can increase users' perception of reality during their training sessions, but its application generates a variety of different combat situations where users can exercise their skills against the same population of Shootwell ICs multiple times. Automated IC behaviors, whether active or reactive, are affected by the values of their personal attributes. Attribute parameterization of Shootwell IC actions is combined with an element of randomness to add variability to IC behaviors. Some examples of parameterized variability are:

- Upon hearing the sound of footsteps on a tile floor in an adjacent room, an experienced combatant turns towards the sound and sprays fire through the wall; an inexperienced combatant does not realize the sound indicates an opportunity to attack.

- When an experienced combatant detects the voice of a user's avatar (via headset microphone) outside the door of a room it is hiding in, it does not immediately fire through the door. Instead, it concentrates on the door, ready to fire, but after a few seconds it tries spraying fire at the wall adjacent to the door in an attempt to hit user avatars that might be gathering there.
- When suddenly confronted by an enemy, a nervous character wildly sprays fire in the general direction of user avatars. A calmer character takes a brief moment to aim accurately before firing a burst.
- Upon seeing a grenade slowly tossed into a room, a timid character runs to a cover position to avoid the blast, while a bold one attempts to throw the grenade back out of the room.
- If a nervous combatant hears many different gunshots or explosions over a period of time, it exhibits fear by dropping its weapon and surrendering to a visible user avatar. If an experienced character hears one grenade detonation following another after just a brief interval, it retreats to a cover position in anticipation of further grenade attacks.

The type and magnitude of a given character's attributes affect the selection, timing, and interpretation of actions performed. Behavior execution is also mildly randomized to mimic the dynamic nature and complexity of real-time decision-making. The amount of random variability is distributed across a range that is relevant to a population of auto-ICs. If a character and other auto-ICs in its population have very high shooting accuracies on average, the character's deviation from well-placed shots will not be entirely erratic but will fall within a normal distribution around its population's average.

We want Shootwell's ICs, like humans, to learn and generalize from previous engagements. This could enhance realism, and it adds repeated usability to a scenario because of Shootwell ICs' evolving combat behaviors over repeated training runs. During execution, a player profile is collected for feedforward input to virtual combatant decision-making logic in future runs. Users' known patterns or habits are avoided by auto-ICs. Where a user's actions are deemed unsuccessful, e.g., after failing to detect an enemy combatant in a closet, the combatants more often act in ways that target these weaknesses. Whether a user is successful in clearing a particular room or not, virtual combatants also try to force users to follow prescribed practices of where/what/how to perform by "surviving" user techniques that deviate from standards. Combatant character profiles themselves can be morphed by adaptation to user inputs.

4. TECHNICAL DESIGN

We have described some of the human-like things that Shootwell ICs can do. The behaviors of the automated ICs in the virtual environment are scoped by their physical surroundings and parameterized by their unique personal characteristics. Variability of the same or similar characters is accomplished by applying a small degree of randomness and, optionally, feedforward input from the actions of users in previous training runs.

The decision-making of Shootwell ICs applies to two hierarchical levels: "atomic actions" and "behaviors." An action is considered atomic by our definition if it suits the largest subset of enemy characters that will perform that action identically in the same situation. An example of a fairly common action is walking from one place to another. Given any uninjured character with any personality, its movement from some starting point to some ending point will follow the same path as another such character. We define a "behavior" as an ordered arrangement of atomic actions. The behavior of "responding to user attack" can consist of quite different sequences of atomic actions between two characters with differing auto-IC profiles. One character may seek cover and return well-placed, effective weapon fire. Another character may discard its weapon and surrender.

4.1 Action States

Surrounding Shootwell ICs' artificial intelligence are state loops that guide automated ICs to seek new behaviors when idle, rather than waiting for a triggering event such as voices or weapon fire. These are the primary stages that auto-ICs follow to engage user avatars in direct combat:

- Scan for target
- Acquire target
- Choose weapon
- Ready weapon
- Choose firing mode (burst, single shot)
- Select firing position
- Assume posture (prone, standing)
- Fire weapon

Lower-level actions and planning may temporarily or permanently interrupt a default combat state. Such actions may be associated with stimuli, randomness, or conflict resolution.

Ordered atomic actions compose complex combatant behaviors in a virtual setting. Action nodes and predicate links are authored via flexible, hierarchical script components using the SimBionic AI middleware toolkit.

4.2 SimBionic Scripts

Activity scripts are authored and executed by SimBionic, an AI middleware toolkit produced by Stottler Henke Associates, Inc. SimBionic's authoring tool surfaces parts of Shootwell IC programming logic in a Graphical User Interface which makes it accessible even to non-technical parties for review.

In the SimBionic editor window shown in Figure 1, character action is specified with flow-chart diagrams. Primitive actions are represented by rectangles, calls to other script components by boldfaced rectangles, conditions by ovals, and control flow by line segments. Polymorphism features of SimBionic allow the same named action to be implemented differently for diverse subsets of virtual populations. Determining which action to execute is done at runtime by processing a character's attribute and state settings.

SimBionic utilizes behavior transition network (BTN) concepts to provide intelligent agent modeling. BTNs are generalizations of finite state machines. BTNs have the "current states" and "transitions" of finite state machines. They also can hierarchically decompose, have variables, message other BTNs, and execute arbitrary perceptual or action-oriented code. A large number can run in parallel.

A node in a BTN represents a primitive action, coded in the script engine, which an entity performs at some point during a simulation. The current node of a BTN denotes the activity currently being carried out by the associated entity. A given BTN may have exactly one current node at a time per execution frame. A transition in a BTN is a directed arc

4.3 Script Engine

Atomic actions are combined dynamically to produce a huge number of unique Shootwell IC behaviors. In the runtime engine, each AI-directed character associates with one or more script instances that guide how it will function in the simulation environment. At each time step the engine examines the entity's execution stack to determine the next transition to be followed. Transitions are evaluated for the current node in each execution frame, starting at the bottom of the stack and stopping when an active transition is found. The active transition path is followed, changing the current node and causing the associated action to be performed by the entity. In addition, all execution frames above the one

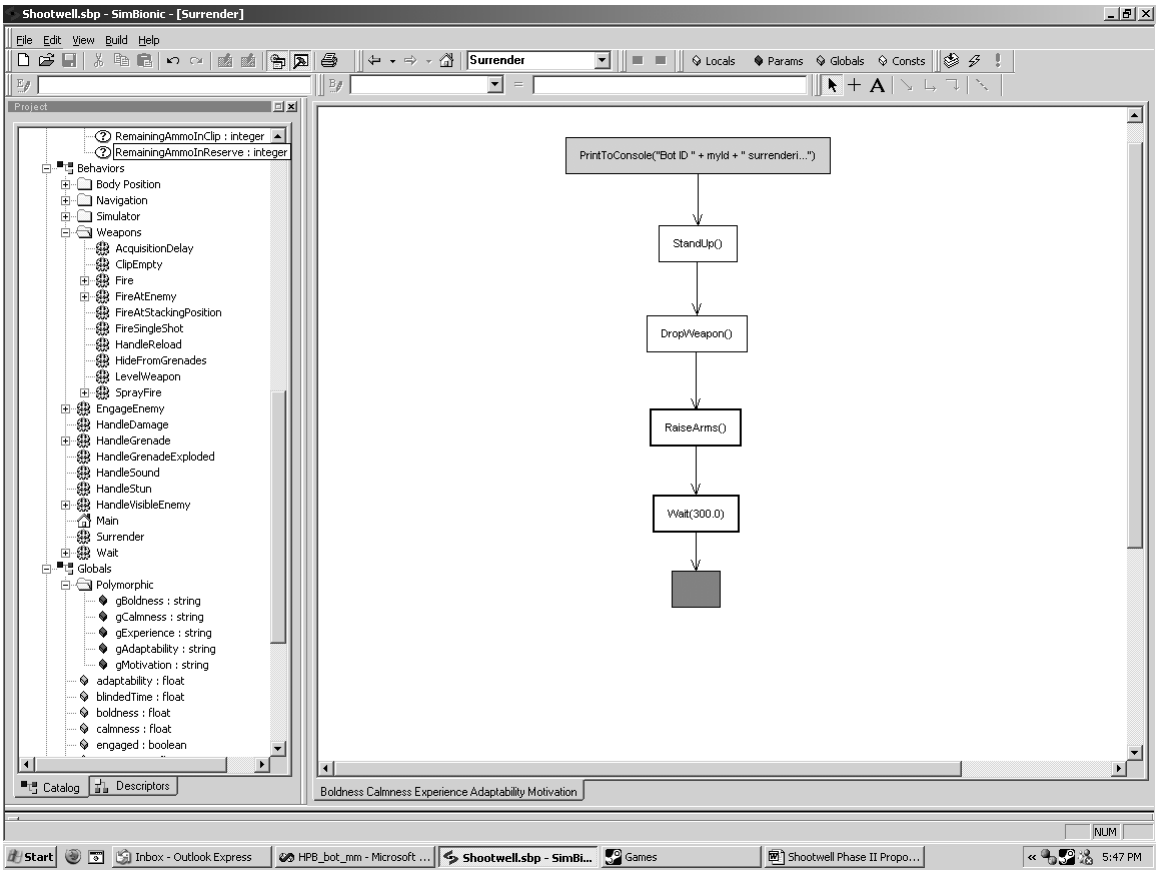

Figure 1. SimBionic Editor

connecting two nodes X and Y, or looping from one node back to itself, and indicating a potential direction of control flow. A decision is associated with each transition. A transition is said to be active if its decision process returns a "true" result, so an active transition means that the BTN should change its current node from X to Y.

As stated, BTNs may be hierarchical. That is, any node in a BTN may link to another arbitrary BTN or set of BTNs. When a node with such links becomes current, execution passes to one of the linked BTNs, which begins at its initial node. Each time this occurs, a new execution frame is put on the character entity's execution stack which maintains the behavioral state at each level of the currently executing BTN hierarchy for that entity. This tiered structure allows for natural decomposition of tasks and abstraction of high-level behaviors.

with the active transition are popped from the stack.

4.4 Dynamic Behaviors

Our character design includes the identification of character capabilities that can allow the execution of realistic virtual combatant behaviors. Shootwell IC capabilities include finding a hiding spot, loading a weapon, acquiring a target, and throwing a grenade. We specify atomic actions for the auto-ICs using the SimBionic authoring tool and runtime engine. To create more complicated "behaviors," we want to "mix and match" these actions in ways that are realistic and meaningful.

Considering the multiple situational and personality characteristics taken into account, pre-defining enough complex behavior sequences to convey realistic combat scenarios is intractable. We get flexible alternatives by

conditionally stringing together parameterized actions dynamically, at runtime. This is the code behind the conditional ovals (predicates) in the SimBionic scripts.

In addition to analyzing characters' attributes and states to select and customize auto-IC actions, Shootwell code processes interactions amongst character attributes.

The matrix in Table 1 illustrates simplified cross-effects amongst attributes. Falling `energy` can decrease an entity's `fitness` while negating the rate of change of its `hydration` due to slower activity. Falling `hydration` can reduce both `energy` and `fitness` states. Since decreasing `hydration` reduces `fitness`, it can accelerate the decrease in `hydration` due to increasing water needs.

Table 1. Attribute-Attribute Effects

	↓ Energy	↓ Hydration	↓ Fitness
Energy		↓	↓
Fitness	↓	↓	
Hydration	↑		↓

Table 2 similarly illustrates the effects of changing state levels on combat abilities.

Table 2. Attribute-Action Effects

	↓ Energy	↓ Hydration	↓ Fitness
Firing Mode			↓
Target Aim	↓	↓	↓

4.5 Conflict Resolution

If an auto-IC is carrying multiple weapons, will it choose the weapon preferable for a given target but with little remaining ammunition, or will it choose an alternate weapon with plenty of remaining ammunition? Conflicts like this are resolved by the explicit hierarchical implementation of differing outcomes. If remaining ammunition were always the most important factor, the ordering of weapon choices based on available ammunition would be represented at a higher level and thus could supersede the optimal weapon choice that was based on target.

4.6 Indeterminism

There is not a direct correspondence between each virtual combatant's attributes and a combat outcome. Most actions have a many-to-one connection from attribute and state values. Furthermore, conditional selection of actions is sometimes gated by a probabilistic likelihood of selection. When action and state transitions include an element of chance, a random number determines which alternative to execute.

Since in most cases it is unlikely that each alternative action will be as likely as any other, a weighting scheme based on Shootwell IC characteristics is used to make certain paths more likely than others. These weights are defined in the conditional code itself. Consider an example where

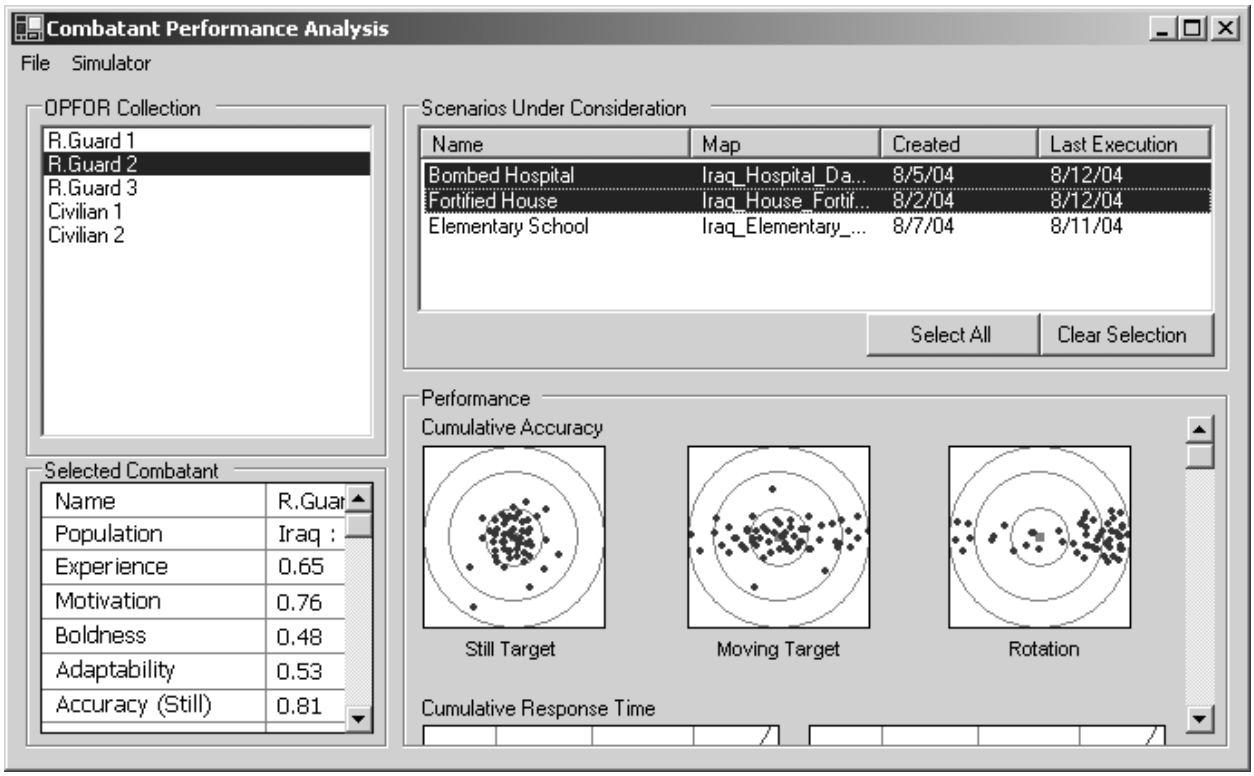

Figure 2. Performance Analysis Tool

weighting applied to the action of "throwing back a grenade" is associated with the boldness level of an auto-IC. A *moderately* bold combatant may not throw back a grenade every time. But, that virtual combatant might be more than 50% likely to make the split-second decision to grab and return an incoming grenade rather than to instantly seek cover.

Customization of actions can also be affected by weighted probabilities. For example, "effective combat skill" is an average value. The implementation of an attack on a user avatar by a Shootwell IC is determined by taking the shooting accuracy of the virtual combatant, setting a range of values around that accuracy, and randomly calculating the weapon's aim point to fall within the targeted range.

Other weighting factors used are health, experience, and situational parameters such as distances, angles, and surprise.

5. VALIDATION

The outcome of training with our virtual combatants is not easily testable and largely subjective. Following are some methods that we feel will give a grade to the performance of Shootwell AI for Individual Combatants in the training scenario. Additional applications of these types of tools will give us latitude to adjust Shootwell IC behaviors.

5.1 Subject Matter Experts

The relatively unbounded scope of a simulated training environment makes comprehensive evaluation of combat behavior correctness impractical, if not impossible. Because perceived realism is our primary concern, we will take advantage of the experience of human experts to obtain a qualitative evaluation of the gross behavior of combatants in representative situations.

5.2 Turing Test

A Turing test might be a useful indicator of the realism in a training combat scenario. Note that advance consideration of this test plan during technical design is necessary to either provide direct human control of enemy Individual Combatants or to avail human-controlled avatars of the clothing and equipment to imitate automated ICs. The ManSIM application, to which we will interface Shootwell components, supports multiple avatars under human user control. We can use the multi-user mode to implement our test model:

1. Populate a scenario with a combination of automated ICs and human-controlled ICs. Human-controlled ICs need to be visually indistinguishable from auto-ICs.
2. Reveal to users the total number of enemy ICs they will encounter during the simulation.
3. Following the training session, solicit the human testers' opinions as to the ratio of automated to human-controlled entities. (Ideally, each avatar would be accompanied by a unique identifying icon to allow human/not-human identification per character.)
4. Repeat for each fighting scenario and for various ratios of virtual combatant avatars and human-controlled avatars.

The test passes when the average percentage of correct user guesses is less than a pre-set quality threshold. A risk that could invalidly *increase* the number of correct identifications by users is the presence of irrelevant artifacts of controller input. This can impose a development requirement to restrict IC actions to match only those actions that are possible to express through controller input.

5.3 Performance Analysis Tool

Though qualitative judgments of Shootwell ICs' behaviors are constructive for initial design and troubleshooting, quantitative performance metrics could enhance fine-tuning of virtual combatant populations. A statistical analysis tool (see Figure 2) displays data about user and virtual IC weapon-firing accuracies, response times, and other simulation-accessible parameters of interest.

Statistics collected in a particular map area that deviate too much from average data highlight a scenario that deserves tuning attention. Either difficulty balancing is in order, or a bug in virtual combatant behavior may be indicated.

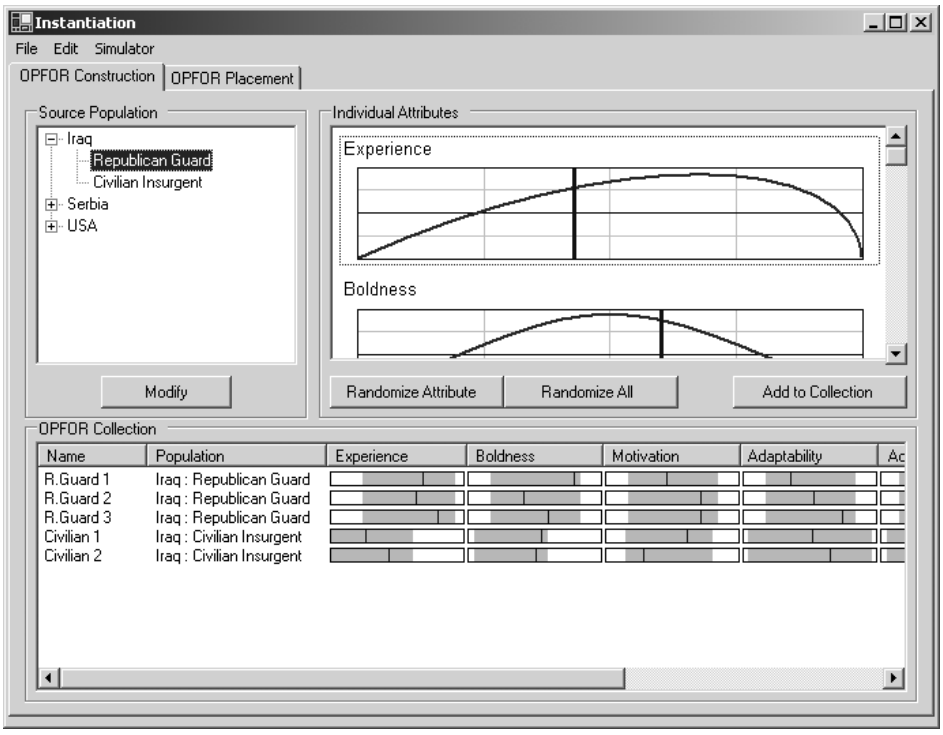

Figure 3. Population Definition Tool

Populations of Shootwell Individual Combatants can be examined for their adherence to an expected range of competence. Individual virtual opponents that are equally competent may be grouped into training levels according to their average performance, so as to increase the difficulty for the user as he or she advances.

The tool and analysis can also be used as part of an After Action Review in which an instructor discusses performance with a student user.

5.4 Population Definition Tool

The population definition tool (see Figure 3) could dramatically increase the usefulness of our AI system. The tool will define attribute values for a wide variety of adversary populations. Models might be defined for Iraqi insurgents or Iranian conscripts, for example, that then could be instantiated in simulated combat environments. The population definition and editing tool will help manage the creation and modification of these populations, allowing the specification of behavioral parameters to closely approximate those that have been associated with a particular group in the real world.

To facilitate the creation of a realistically non-uniform population, users will define statistical distributions for auto-IC attributes, capturing a range of observed behaviors across a population. Distribution definition will be carried out either parametrically by the definition of mean and standard deviation values, per attribute, or graphically by the creation of distribution curves. These curves can be associated with an unlimited number of source populations, which can themselves be nested to form hierarchical definitions of population properties that inherit from one another. The tool will provide distribution templates like the Gaussian as well as facilities for algebraic definition of custom distributions.

The resulting statistical models will be used to generate a set of Shootwell IC profiles for a force whose characteristics are probabilistically chosen from the tool models and representative of their populations.

6. TUNING

Tunable features can afford a spectrum of flexibility to customize battles and their automated characters. They can extend the relevance and life of training environments.

Our initial attention to tuning is to maximize the relevance and volume of feedback that Subject Matter Expert (SME) evaluators of Shootwell applications can provide to us. We will use their feedback to improve the realism of Shootwell combat behaviors in subsequent versions. For efficiency in experimentation with attribute values, Shootwell will be able to reload virtual combatant profiles on demand, at runtime. We also will implement behavior "dials" in a simple GUI front-end to Shootwell that can agitate profile data. Two dials will modify the data in different ways. One will increase or decrease the values of all attributes. The other dial will randomly modify all attributes. An additional runtime option will serve as the Shootwell equivalent of a "screenshot": it calls a method to write new attribute values over the stored IC text data. Evaluators will be able to accompany their descriptive comments with a "dial setting" or new IC profile data that they have observed to result in either the correct or problematic IC behaviors they describe.

Later versions of Shootwell will increase complexity in the processing and execution of automated IC combat behaviors, with emphasis on maximizing the training value of the virtual environment for student users. Shootwell components will be able to run in "IC training" mode to reinforce or weaken automated IC choices according to their successes in combat engagements. Since both relevant and irrelevant parameters (attributes) would be considered during each training epoch, a lengthy learning time or even over-generalization of auto-IC attributes would be possible. However, if a human user were skillfully engaged, such training over a number of battles could result in positive learning for the Shootwell ICs.

The correlation between virtual combatant actions and successes (or failures) could be strengthened by using the Turing test validation capability described above. A human avatar could impersonate a Shootwell AI character to "teach it" the correct way to act. Deliberate human control of both competitors (opponents) in staged battles could limit specious activity and thus increase the percentage of relevant input to a post-processing training pass.

7. SUMMARY

We are attempting to create the most realistic human-combat behaviors possible in a virtual environment. Shootwell artificial intelligence technology controls individual enemy characters in a first-person combat-training application.

Our automated combatants need to behave as realistic humans in battle. We have selected some features like weapon skills, physical health, and personality traits like "boldness" that could affect personal choice and performance. We have solicited feedback from experienced SMEs to define what and how certain personal features might shape the way a combat engagement plays out.

Our technical design associates each Shootwell IC and population of ICs with text-based profile data that assigns unique values to select traits. The profile data is used to satisfy conditional logic in top-level hierarchical scripts assigned to ICs. In addition, probabilities weight certain code paths an IC might pick when multiple choices would be reasonable. A random number is used to select a path probabilistically. Following an action path can change an automated IC's dynamic states (e.g. injury level) as well as its original profile data (e.g. combat experience). Such a change can modify an automated combatant's future behavior.

We continue to implement the system described here. Much of our current effort is dedicated to completing a flexible framework that supports ongoing extension to auto-IC capacities and complex behaviors. The framework includes customizing a Shootwell High Level Architecture Federate; coding of the SimBionic interface, conditionals, and primitives; configuration of the user-facing visual environment; and extension of our analysis (e.g. spatial analysis for cover selection) library to evaluate physical surroundings.

To validate our approach and implementation, we have solicited design feedback from SMEs who will also evaluate our functional system and tools. We have several version releases of Shootwell planned to iteratively collect and

incorporate feedback to augment the realism of its controlled combatants. After refining strategies to control Shootwell ICs defending building interiors, our research and development will expand the scope of their autonomy. Future efforts include communication and team navigation amongst virtual combatants, offensive tactics, and outdoor infantry combat.

8. REFERENCES

[1] MCWP 3-35.3, Military Operations on Urbanized Terrain (MOUT), US Marine Corps.

[2] Ceranowicz, A. and Torpey, M. Adapting to Urban Warfare, In Proceedings of the Interservice/Industry Training, Simulation and Education Conference. I/ITSEC, Orlando, FL, 2004.

[3] Fu, D. and Houlette, R. T. Putting AI in Entertainment: An AI Authoring Tool for Simulation and Games. IEEE Intelligent Systems, Vol 17, No 5, p81-84, 2002.

[4] Henninger, A., Jones, R., and Chown, E. Behaviors that Emerge from Emotion and Cognition: Implementation and Evaluation of a Symbolic-Connectionist Architecture, In Proceedings of Autonomous Agents and Multiagent Systems. AAMAS, Melbourne, Australia, 2003.

[5] Jones, R. An Introduction to Cognitive Architectures for Modeling and Simulation, In Proceedings of the Interservice/Industry Training, Simulation and Education Conference. I/ITSEC, Orlando, FL, 2004.

[6] Laird, J., and van Lent, M. Human-level AI's Killer Application: Interactive Computer Games. In Proceedings of the National Conference on Artificial Intelligence, 2000.

[7] Livak, T., Hefferman, N. T., and Moyer, D. Using Cognitive Models for Computer Generated Forces and Human Tutoring. In Proceedings of the Conference on Behavior Representation in Modeling and Simulation. BRIMS, 2004.

[8] MacKenzie, D., Arkin, R.C., and Cameron, J. Multiagent Mission Specification and Execution. Autonomous Robots, 4(1): 29-57, 1997.

[9] McDonald, B., Weeks, H., and Hughes, J. Development of Computer Generated Forces for Air Force Security Forces Distributed Mission Training. In Proceedings of the Interservice/Industry Training, Simulation and Education Conference. I/ITSEC, Orlando, FL, 2001.

[10] Middleton, V., D'Enrico, J., and Christenson, W. Simulation of Suppression for the Dismounted Combatant. In Proceedings of 5th Conference in Computer Generated Forces and Behavior Representation. Orlando, FL, March 1997.

[11] Reece, D. A. and Kraus. "VIRTE Demo II Final CGF Systems Analysis," SAIC, Orlando, FL, 2003.

[12] Reece, D.A., Kraus, M.K., et al. Tactical Movement Planning for Individual Combatants, Simulation Interoperability Standards Organization. Orlando, FL, 2000.

[13] Stottler, R., Lackey, S., and Kirby, J. B. Formalized Behavior Models for MOUT OPFOR Individual Combatant Weapon Firing. In Proceedings of the Interservice/Industry Training, Simulation and Education Conference. I/ITSEC, Orlando, FL, 2004.

[14] Summers, J. D., McLaren, B. M., and Aha, D.W. Towards Applying Case-Based Reasoning to Composable Behavior Modeling, In Proceedings of the Conference on Behavior Representation in Modeling and Simulation. BRIMS, 2004.

[15] UO FACT, "Modeling Individual Combatants in Urban and Open Terrain," White paper, 28 June 2005: https://www.moutfact.army.mil/frameset.asp?sec=whitepapers&submenu=whitepaper_directfire

[16] Wray, R. E., Laird, J. E., Nuxoll, A., and Jones, R. M. Intelligent opponents for virtual reality trainers, In Proceedings of the Interservice/Industry Training, Simulation and Education Conference. I/ITSEC, Orlando, FL, 2002.