

DEFEATING THE AUTHORIZING BOTTLENECK: TECHNIQUES FOR QUICKLY AND EFFICIENTLY POPULATING SIMULATED ENVIRONMENTS

Bill BLANK
DI-Guy Product Manager
Massachusetts, USA

Alex BROADBENT
Adam CRANE
Gedalia PASTERNAK
DI-Guy Senior Engineers
Massachusetts, USA

ABSTRACT

Populating virtual environments with realistic human characters is critical for most modern simulations. The increasing importance of human simulation is a result of a number of factors: 1) The increasingly crucial role individual human entities play in the battlefield, 2) Urban environments where enemy targets are embedded within large civilian populations, and 3) Renewed strategic emphasis on “winning the peace by winning the hearts and minds of the local inhabitants.” Terrains that do not include realistic human characters as part of the landscape do not completely fulfill their mission of providing settings and conditions for training and analysis that duplicate critical theatre conditions that impact mission success.

An important limiting factor in the applications of humans to battlefield landscapes is what we identify as the Authoring Bottleneck. In short, if Subject Matter Experts do not have efficient techniques for authoring human performances that are both user friendly and flexible, simulations that fulfill the hardware, geometry content, and motion content requirements to populate virtual battlefields will still remain empty “ghost towns”.

In this paper we introduce the concept of the Behavior Authoring Pyramid, a series of four authoring techniques (Action, Path, AI Base Behavior, and AI Mind) which together form a powerful paradigm to defeat the Authoring Bottleneck.

INTRODUCTION

Over the past ten years, we have researched and implemented a number of techniques for authoring human performances. The goal has always been consistent; “How can we empower Subject Matter Experts to create compelling human-oriented scenarios. Above all else, how can we give the SME authoring tools whose focus is efficiency, where the SME can get his work done in a reasonable amount of time, under budget, and without requiring software programmer intervention?”

In the early years, our techniques focused on the authoring of individual character performances, with the final scenario being the sum of those individual performances. In recent years, our research has shifted to authoring performances with hundreds or thousands of characters. Earlier techniques did not always extend well to these larger population authoring requirements. Therefore we needed to develop

new techniques to meet these high entity count challenges. The new techniques emphasized two components: 1) terrain-aware “agent” characters featuring a high degree of autonomy and using artificial intelligence to make decisions, and 2) dealing with aggregates of individual characters as single entity “crowds” that could be populated, modified, and simulated as a single entity.

What emerged from these various efforts was a coherent set of authoring solutions that we refer to as the Behavior Authoring Pyramid. The pyramid features increasingly sophisticated techniques for authoring human performances that build upon each other. Each technique has advantages and disadvantages; no single technique alone solves all human simulation challenges. But together, the sum of these solutions is a dynamic authoring environment where SMEs can select different techniques from the pyramid to match the goals and requirements of the entities in his simulation. These solutions can be mixed within a single scenario where some characters may be authored via one technique and

interact with other characters authored via another.

BACKGROUND

The military Visualization and Simulation (VizSim) community has seen a dramatic shift in attitudes towards human simulation in the last two decades. Once the exclusive domain of high-flying aircraft and tank trainers, these simulations initially had no humans. As technologies improved, human simulation began to elbow its way into the VizSim community; simulations started featuring a small number of humans scattered about the scene, either playing the role of a few “critical individuals”, a squad of simulated infantry, or visual background clutter. Today, in the light of challenging peacekeeping assignments, anti-terrorism, and asymmetric warfare, there is widespread recognition that the power and influence of the individual in the battlefield has never been higher. Therefore simulated humans, numbering in the hundreds or thousands, modeling civilian populations, friendly forces, oppositional forces, and mixed populations are a critical component in the accurate simulation of real world situations, particularly urban operations.

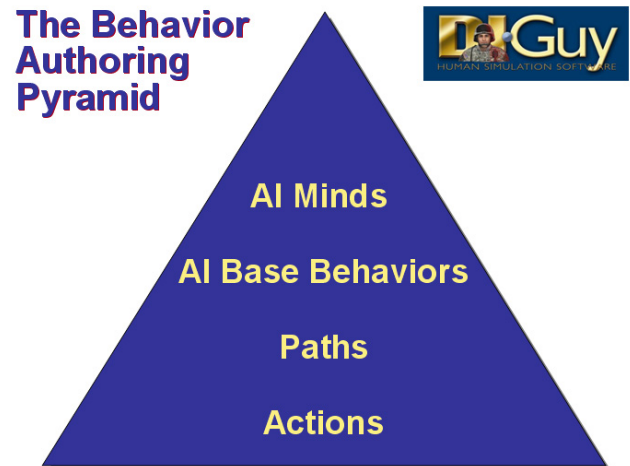
Simulations featuring large numbers of realistic looking and behaving human characters are therefore becoming critical components of virtual training activities, planning and analysis applications, and mission success. Without accurate and affordable human entities, simulations will fail to provide the training and analysis needs of the armed forces, emergency response, and law enforcement communities. For a number of scenarios, in fact, the paradigm has been turned on its head: Instead of human simulation as an afterthought, a “nice to have” of background clutter for vehicle simulations and realistic terrains, the human entities and their behavior are themselves becoming the focus. Examples include cultural training, leadership training, law enforcement shoot/don’t shoot arrest trainers, crowd control analysis, security, and societal modeling.

Populating urban environments is a challenge on many levels. Unlike simulating a specialized piece of equipment, all of us are experts on how humans look and behave. If a human model is of poor visual quality, it immediately draws the eye and distracts from the immersive experience aimed for in training. If the character moves unnaturally or makes behavior decisions that seem unrealistic or unlikely, the user is distracted. If the virtual human characters do not react to ongoing events in a simulation or their reaction is unbelievable, distraction is again the result.

But it is equally important to note that the lack of humans in a simulation can be equally distracting. All of us have seen impressive simulations with detailed terrains, sophisticated

and accurate building models, vegetation, and realistic vehicle models, but are eerily unpopulated; these terrains register to users as highly synthetic ghost towns.

THE BEHAVIOR AUTHORIZING PYRAMID



The above image shows the Behavior Authoring Pyramid, the core idea being transmitted in this paper. The four different authoring techniques, Actions, Paths, AI Base Behaviors, and AI Minds compose the pyramid. In the following sections, we will describe each technique and discuss its advantages and disadvantages. Just like a pyramid, each level builds on the foundation created by the previous levels, achieving a higher objective.

Behavior Authoring Pyramid Technique #1 - Actions

Action authoring is the most straight-forward authoring technique. Using this technique, the author commands the character to perform an action, such as “run”, “walk”, “sit”, or “prone_aim”.

Underlying software should take care of performing the actual animation of the character, relieving the author of the burden of worrying about joint angles, velocities, etc. Our experience is that the best animations result from performing motion capture of skilled individuals actually performing the tasks vs. skilled animators building motions from scratch. Another important consideration for the underlying software is to perform realistic, eye-pleasing transitions between actions. For example, a soldier character commanded to perform a “prone_aim” action when he is performing a “run” should automatically slow down, stop, kneel down, and go prone, (or alternatively, perform a combat dive). In both cases, the transition motion is as important as the base action motion, and needs to far more than a simple linear interpolation between joint angles; indeed, the transitions

should be considered and recorded during the motion capture session used to obtain the base action motions. Finally, the underlying motion engine should accumulate position, and where appropriate, orientation of the character.



Figure 1 - A soldier performing a "run" action is commanded to "prone_aim" and executes a realistic transition.

Animation Blending

In addition to the main action being commanded by the author, animation blending is used to layer other sub-behaviors on top of the base behaviors. Examples include:

Gestures – gestures are best thought of as sub-actions that are applied over an action. Typically gestures are associated with verbal behavior (e.g. hands to side and head cocked to one side when saying “I don’t know”), but gestures can also be signals, sign language, head nodding, pushing buttons or operating controls, etc. Authoring gestures typically involves specifying a particular gesture, its intensity, and number of repetitions. The underlying motion software should handle mixing the gesture to the ongoing base action.

Aiming – aiming typically involves having motions recorded with particular weapons. Aiming a pistol is different from aiming a rifle, which is different from aiming a javelin. The author should only have to specify very simple parameters about where to aim, either via azimuth and elevation, or via a particular character or target. The underlying software should understand the weapon being used and invoke the proper motions to perform the aim. Note that the aim space is typically a continuous two-dimensional space (azimuth/elevation) where the character should be able to realistically aim his weapon anywhere within the azimuth and elevation ranges.

Weapon Firing – once aimed, the author can tell the weapon to be fired, with the underlying software handling the fire motion, special effects for weapon flash, and even projectile trajectory and hit detection.

Gazing – gazing is an important function as it indicates interest and focus of a virtual character. Similar to aiming, the author should be able to simply identify a gaze target.

Expressive Faces – showing emotional content, blinking realistically, or lip-syncing to sound files, commanding expressive faces as sub-actions to be layered on top of the base action performance is often handled at the Action level.



Figure 2 - A cultural trainer features gestures and expressive faces as critical parts of the human performance.

Hand articulation – this is really a subset of gestures, where the author may want to command how the arm and hand should move as a layer on top of the base Action, but perhaps in a programmatic way instead of a pre-captured gesture.

Guides and Formations – useful when the author wants a character’s position to be governed by other external data (e.g. – incoming intermittent remote positioning data or formational positioning based on a leader character’s position).

Action Authoring Technique - Advantages

The advantages of Action level authoring are:

- 1) Simplicity
- 2) Direct control of character
- 3) Action layering

Action authoring provides an important layer of abstraction to the SME, removing the need to perform direct joint control or create animations.

Action Authoring Technique – Disadvantages

The disadvantages of the Action techniques include:

- 1) Lack of topological information – with actions, you can only specify a starting place for a character and then accumulate position as actions and orientation changes are applied.

- 2) Lack of “future” conceptualization – with actions, one is typically commanding a character what to do right now.
- 3) Requires either constant human intervention (to tell the character what to do now) or home-brewed behavior algorithms to command from a higher-level what actions the character should be performing. This last item is exactly the sort of inefficiency that results in the Authoring Bottleneck.

Behavior Authoring Pyramid Technique #2 - Paths

Path authoring was our first significant step to empower SME authors. With path authoring, the author places a spline path on the terrain composed of two or more waypoints that define the shape of the path. The path does more than just control how the character moves topologically on the terrain. It also commands the character to perform actions and evaluate logic conditions as the character progresses along the path.

Authoring of paths, while possible through a programming API, is designed for point-and-click GUI authoring. In the 3-D Window the author clicks and drags on a loaded terrain to “blitz” in a new character. The author can then click in more waypoints to define the path shape and edit those waypoints by dragging them on the terrain or modifying the curvature of the spline as it moves through a waypoint. The author then typically clicks in action events (we call these action beads, as they visually resemble beads on a necklace) which command action changes to the character, so that, for instance, a character that starts with a walking action will start to run when it reaches a “run” action bead set some meters down the path.

The types of beads that the SME author can add to the path are not limited to actions. Aim beads and gaze beads also enable the SME to create layered performances. Decision beads use a pulldown menu system to enable the SME without programming to add logic to the path. For example, the SME can use this programmer interface to tell a soldier to aim and fire his weapon if there is an enemy combatant within a set distance of the character. Or set an expressive face based on a condition or apply a gesture. The conditional tests do not need to be just a one-time event; the bead can be specified to last a certain amount of time, with the conditional testing applied periodically as determined by the SME.

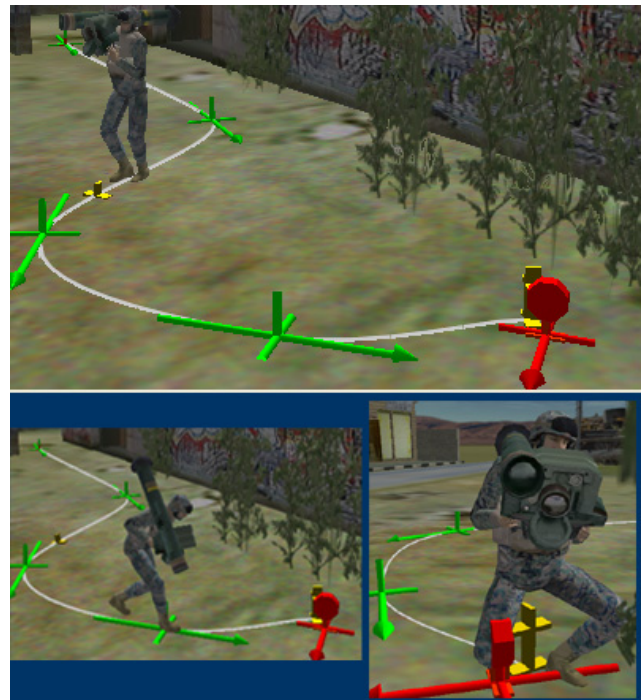


Figure 3 - A soldier with a javelin moves on a path defined by five waypoints. He walks until he reaches the yellow action bead which commands him to "walk low". When he reaches the next action bead he performs "kneel aim".

SMEs with PERL scripting skills can transform decision beads into script beads, and extend the decision logic and make API function calls to the character, the scenario, or even other characters in the simulation.

Multiple paths can be associated with a character and can be defined in world coordinates or in the local coordinates of the character. This means that a complex sequence of actions defined on a path, such as refueling a vehicle, can be created as a local path which can then be triggered from a main path the character is on when a refueling event occurs. In this way, complex logic and a rudimentary artificial intelligence can be built up to make sophisticated performances that react to changing events in the scene.

Path Authoring Technique - Advantages

The advantages of Path level authoring are:

- 1) Generally uses no programming – extremely SME friendly.
- 2) Excellent for detailed, exact performances.
- 3) The lack of topological and temporal behavioral

IMAGE 2009 Conference

control missing from Action level authoring is resolved.

Path Authoring Technique – Disadvantages

- 1) Does not extend well when many characters are required, as each character requires its own path authoring.
- 2) No collision detection.

Behavior Authoring Pyramid Technique #3 – AI Base Behaviors

AI Base Behaviors were developed to resolve the poor extensibility of the Path and Action authoring methods. In this authoring method, the SME author selects a crowd profile which defines the number of characters to put in the crowd, the various appearances of the constituent members of the crowd, and then paints in a population region on the terrain. The agents who make up the crowd are inherently different from the directly controlled characters of the Path and Action authoring methods. While those characters depended on the SME author to determine their position and behavior at any given moment, the agents are independent autonomous characters who determine by themselves when and how to move.



Figure 4 - Soldiers use an "Attack AI Base Behavior" to move to kill an opfor who has wounded one of them. For this soldier crowd, the opfor is their "focus character". The agents are controlled at a more abstract level by assigning an AI Base Behavior to the crowd. Example AI Base Behaviors include:

- Wander – agents perform a stationary action for a period of time and then randomly determine a new position in the world to move to. The agents then navigate to that new position through the terrain while avoiding obstacles and other characters.
- Mingle – similar to wander, characters that mingle will select points near other characters of their crowd.
- Travel – a path is associated with the crowd and characters use that path as a loose guide on where

to move.

- Flee – moving away from a point or character, agents navigate through the terrain.
- Follow – agents follow a particular entity.
- Attack – agents not only follow a particular entity, they will aim and attack when within range and have line-of-sight.

AI Base Behavior Authoring Technique - Advantages

The advantages of AI Base Behavior level authoring are:

- 1) Generally requires no programming – extremely SME friendly.
- 2) Excellent for populating simulations with lots of characters.
- 3) Characters perform collision detection and avoidance with terrain and other characters.
- 4) Agents perform smart terrain-aware navigation.
- 5) Agents do something a little different every simulation run, enabling natural variation in repeated runs of a scenario.



Figure 5 - a crowded Mideast street scene features various pedestrian crowds modeled using AI Base Behaviors.

AI Base Behavior Authoring Technique – Disadvantages

- 1) Lack of direct control is not always desirable – not ideal when a very specific behavior is desired.
- 2) Agents perform their AI Base Behavior indefinitely without external logic to further modify behavior.

Behavior Authoring Pyramid Technique #4 – AI Minds

AI Minds are the “capstone” of the Authoring Pyramid. Each agent blitzed onto the terrain now comes with an independent artificial intelligence brain. This brain defines a State Machine via an accompanying Lua script. (Lua is a lightweight scripting language popular in the video game industry which we use due to its excellent performance characteristics.) Typical states in an AI Mind might be patrol, engage enemy, retreat, etc. As events in the scene

occur and messages from the operator and/or messages from other characters in the scene are received, these events and messages can trigger transitions to new states in the AI Mind state machine. Each state in the State Machine can leverage authoring methods from lower in the pyramid.

Example: A soldier starts a scenario in Patrol State. The soldier performs a Travel AI Base Behavior to walk a perimeter. His superior officer may then show up and order him to follow him, switching the character to a Follow AI Base Behavior, or perhaps using a formation from the Action Authoring level. Note that the message from the superior is what triggered the state change from "Patrol State" to "Follow Leader State". While following the leader, a bomb goes off, wounding the officer. The nearby detonation event triggers a state change which uses a Flee AI Base Behavior to move the soldier away initially, but a "help" local broadcast message received by the soldier then triggers another state change to "First Aid State" where the soldier uses a pre-planned local path oriented to the victim's location where the soldier then performs a series of first aid Actions.

that).

AI Mind Authoring Technique - Advantages

The advantages of AI Mind Behavior level authoring are:

- 1) Non-programming SMEs can blitz in smart characters featuring AI Minds – no muss, no fuss.
- 2) Programming SMEs can extend and build on the AI Minds.
- 3) No external logic is required to effect state changes.

AI Mind Behavior Authoring Technique – Disadvantages

- 1) Like AI Base Behaviors, the lack of direct control is not always ideal – but because the Path and Action authoring methods can still be accessed, this negative is largely minimized with AI Mind characters.
- 2) User-extended AI Minds may create performance problems when sufficient attention to performance efficiency is neglected.

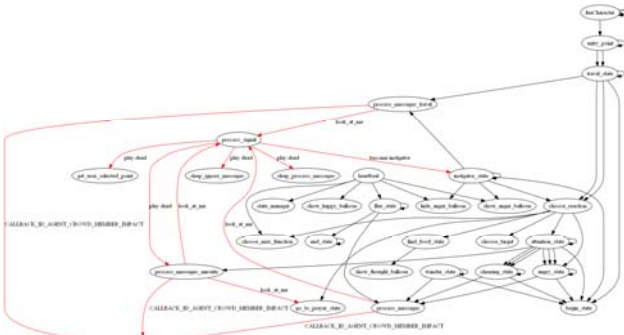


Figure 6 - A DI-Guy AI Mind State Machine graph – auto-generated via GraphViz©.

Lua was chosen as the language for the AI Minds due to its excellent performance characteristics, popularity for similar use in the gaming community, and extensibility without need for re-compile. The AI Minds use a class architecture, so that AI Mind "simpleSoldier" and "simplePedestrian" are sub-classed from AI Mind "simpleCharacter", while AI Mind "soldierSniper" is sub-classed from "simpleSoldier".

It is important to note that non-programming SMEs can simply place and use characters in their scenarios without needing to program; in these cases the SME is using the minds already developed. But it is powerful to know that the SME has access to the Lua source code for the AI Minds and can extend them to specialized characters if desired (even if they need to get a programmer down the hall to help with



Figure 7 - When a nearby mosque issues a "call to prayer", many pedestrians in this scenario alter their current state to "go to mosque". The graphic thought bubbles ("Time for prayer", "I need to pray") help SME authors understand the current state of their agents.

IMAGE 2009 Conference



Figure 8 - Some "instigator" characters have incited a crowd of pedestrians to be angry (visualized with the white "you should be angry" thought bubbles, resulting in an assault on a nearby American humvee. AI Minds have parameterized emotional states that are affected by events and other characters' actions. Angry pedestrians have red "grumble" thought bubbles.



Figure 9 – The mob alters their state machine to flee as soldiers emerge from their burning vehicle. One of the soldiers appears injured; his internal AI Mind will select new states based on his injury, while his companions may also react and broadcast a message to headquarters for an extraction.

CONCLUSION

With a multi-tier approach to human simulation authoring, such as described in this paper, SME authors are empowered to create compelling scenarios because they match appropriate authoring techniques to the differing characters within their simulations. These human simulations can be used in stand-alone training or populate battlefields and terrains via lifeform servers broadcasting characters into a

Presented at the **IMAGE 2009** Conference
St-Louis, Missouri, July 2009.

distributed simulation.

AUTHOR BIOGRAPHIES

Bill Blank received his B.S. in Computer Systems Engineering from Brown University after attending Phillips Academy Andover. Bill has worked in CAD/CAM and medical simulation and for the past 5 years has been Product Manager of DI-Guy (www.diguy.com). Bill is an expert in physics-based and visual simulation.

Alex Broadbent received his Bachelor of Commercial Art from The American College of the Applied Arts. Alex has extensive experience in television, with a decade of work for CNN, ABC Sports, and New England Cable News. For the past eight years Alex has been the Senior Content Engineer at Boston Dynamics and Project Manager for Digital Biomechanics.

Adam Crane received his BS and MS degrees in Computer Science and Electrical Engineering from MIT in 1993. He has been working at Boston Dynamics since 1994 focusing on software engineering, visual human simulation, haptics, and physics-based robotics and equipment simulation. In 1996, Adam created DI-Guy, the de-facto standard for human simulation in the VizSim community.

Gedalia Pasternak received his B.S. in Engineering from the Cooper Union. Gedalia has worked on a range of Computer Graphics and Simulation software over the last 10 years, from television special effects to Massively Multiplayer Real-Time Games. For the last 5 years he has focused on DI-Guy rendering technology and artificial intelligence solutions for human behavior.