Creating Densely Populated Virtual Environments

Ryan McAlinden, Don Dini, Chirag Merchant, Michael van Lent
University of Southern California—Institute for Creative Technologies
13274 Fiji Way
Marina del Rey, CA 90292
{mcalinden, dini, merchant, vanlent}@ict.usc.edu

ABSTRACT
Few virtual environments are capable of supporting large numbers of autonomous agents (> 5000) with complex decision-making on a single machine. This demonstration depicts such an agent infrastructure set within a game-based virtual environment. The embodied agent framework consists of two primary components: a lower-level navigation layer consisting of commercially-available AI middleware, and a higher-level cellular automata system driven by agent goals, resources and thresholds. The overarching game-based infrastructure consists of these two AI components, along with an ICT-developed perception system sitting atop the Gamebryo rendering engine. The typical number of agents supported on a dual-core CPU with a modern graphics card is ~10,000 rendering at 30 frames-per-second. To support this quantity and level of intelligence several design considerations were implemented, including the use of multiple threads, a clone/sprite-based avatar view, and a dynamic level-of-detail update system. Future work includes distributing the AI mechanism across multiple machines to support numbers of agents a level of magnitude higher than is currently possible.

General Terms
Performance, Design, Theory.

Keywords
Cellular Automata, Games, Massively Multi-Agent.

1. TECHNICAL CONTENT

1.1 Background
The embodied agent infrastructure demonstrated here uses a 2-tier AI approach that is integrated with the Gamebryo renderer and several other ICT-developed components. The first tier contains AI middleware (AI.Implant from Engenuity Technologies) that is responsible for all pathing and navigation for individual agents. AI.Implant operates off of a navigation mesh representation and is solely used for calculating and executing agent paths. The middleware runs in a separate thread from the rendering engine to minimize negative performance associated with large-scale pathing and collision calculations. Additionally, a dynamic update mechanism is used to control when agent updates are made to further minimize adverse performance. For example, agents that are located further away from the camera or outside of the view frustum are updated less often than those unoccluded and in close proximity to the viewport.

AAMAS '07, May 14-18, 2007, Honolulu, Hawai'i, USA.

The second, and more complex AI component is the cellular automata resource agent (CAR Agent) system developed by ICT, which is a lightweight mechanism for simulating an urban population. In order to support the maximum amount of realism, each agent is individually driven, thus allowing complex behavior to emerge. Achieving realistic aggregate behavior through the interaction of individual agents has been demonstrated in a wide variety of contexts such as realistic panic models [4], as well as other agent models using cellular automata's (CA) [1]. As individuals, the CAR Agents form a realistic population through two mechanisms. First, they seek out and interact with resources in an urban environment (resource management). Second, the population itself is given a personality and shared history by means of a CA-like mechanism. The CA component allows the propagation of arbitrary knowledge, such as a group of agents reacting to a nearby explosion through second and third order effects.

Sitting between these AI components is publish/subscribe middleware developed by ICT for agent perception. The Agent Perception Broadcast Interface (APBI) relies on embedded contextual information in the environment (called annotations) to pass to agents for use in their decision-making process. The world is marked up with these annotations by a human user to reflect information about the environment beyond the traditional polygons and textures. For example, a structure may be labeled a *restaurant* that serves *middle eastern* cuisine, and by using APBI information such as this can be searched and returned to the agents during runtime. This approach is inspired by *The Sims* which embeds information about how a character can interact with an object (termed an affordance) and in-turn how that interaction may affect the character's model of the object [2]. These affordances are a derivative of J.J. Gibson's affordance theory [3], which argues environmental surroundings convey to humans certain information that directly maps to their decision-making process. One unique feature of our approach is the embedding of cultural information directly in the environment that is also filtered through APBI for use by the AI. This introduces the notion of culturally-induced environmental effects on human behavior, which are modeled as cultural annotations/affordances (i.e., the socio-economic, political, ethnic, historical, episodic, demographic, religious… information associated with an agent's environment). It has been shown that

978-81-904262-7-5 (RPS)

various layers of culture can significantly affect how individuals and groups of individual organize and behave, particularly when their primary motivations are not solely derived from a particular agenda but instead from a diversity of in-groups [5]. Our approach is to embed cultural information within the virtual environment rather than representing this information in the agent's knowledge or memory. The implementation mechanism for such a feature has the user "painting" the ground-plane of the environment pre-runtime with colors that are mapped to a particular cultural affiliation.

Another unique characteristic of this approach is the support for dynamic annotations and affordances that can be modified during runtime. This adds to the environment (and in-turn the agent) the notion of events or history that may alter the AI's behavior. For example, if a faction of AI agents from one political affiliation overtakes an area of town that was previously controlled by another political affiliation, the regional annotation for these areas can be dynamically recalculated to reflect this. This allows episodic or historical information to be embedded in the environment that can then be used by the AI.

1.2 Demonstration Details

The demonstration begins with the initialization of ~8000 autonomous agents randomly assigned to locations on the navigation mesh (which is approximately 4km² and contains ~400 buildings). The environment (i.e. terrain) has been previously modeled, textured, and annotated by an artist, which is cached locally within APBI. The agents are each initialized with a routine set of daily goals (go to work, go home, eat lunch, …) and according to the time-of-day will trigger these goals accordingly inside of the environment. The cellular automata feature becomes pronounced when knowledge of a significant event (e.g. a local food resource becomes empty) alters an agent's behavior, which then influences the surrounding population. Such information is determined by agents through APBI queries, which return what resources (represented as annotations) can satisfy certain goals. These queries can be made fairly complex such as an agent scanning the world for a place to eat, but adding certain constraints (through the search criteria) to narrow the final list of possibilities to select from (e.g., establishment is in-view, or is located in a part of town where the cultural make-up is of a particular type). After the higher-level decision-making has been performed by the agents, the lower-level atomic actions are passed to AI.Implant for execution (move, gesture, …). Finally, the animation calls are passed to the rendering engine for visualization.

All four major components: Gamebryo, AI.Implant, CAR Agents, and APBI run in their own threads to minimize the impact on the renderer. Additionally, both the visualizer and AI.Implant have a dynamic update cycle which only updates agent positions as needed (depending upon the camera perspective and view frustum). The combination of dynamic levels-of-detail and clone/sprite-based representations has allowed us to substantially reduce rendering as the bottleneck and shift our optimization efforts to the AI. Lastly, throughout the simulation the user has the ability to navigate around (either through a 1st or 3rd-person perspective) surveying aspects of the environment (clicking on agents or structures to see goals, resources, etc). They can also alter resources on-the-fly through a slider that will in-turn modify agent behavior in a particular region/block of town.

1.3 Conclusions & Future Work

When simulating thousands of game-based agents on a single machine the bottleneck often arises from rendering issues, primarily because individual DirectX or OpenGL API calls are made to the graphics card for each agent. We have eliminated this issue by representing a sprite/particle-based view for the agents, which has significantly reduced the load on the hardware. However, the performance load has now shifted to the CPU, specifically from the pathing of thousands of agents on a dense navigation mesh. Games typically allot ~15-20% of the CPU for all AI [6]; for this demonstration we have seen the CPU load increase to 60-70% just for the calculation and execution of these agent paths. As a result, framerate slowdowns become more pronounced as increased numbers of agents are added to the scene. One area of future work includes research on distributing the AI across multiple machines, specifically the navigation layer which has proven to be the biggest bottleneck.

Another limitation of our current approach is the lack of true agent coordination. Currently all agent interaction occurs through embedded environmental information or propagation using the cellular automata feature. We would eventually like to support direct coordination between agents that add an additional level of complexity to their decision-making. For example, a group of agents that comprise a single family may coordinate activities to satisfy certain resources such as one member going to the grocery store to pick up food. Our plan is to use the CAR Agents as a foundation on top of which agents using additional deliberative mechanisms such as this can be integrated.

2. ACKNOWLEDGMENTS

The project or effort described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

3. REFERENCES

[1] Batty, M. *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-based Models, and Fractals*. MIT Press, 2005.

[2] Cass, S. Mind Games. *IEEE Spectrum, 39,* 12 (Dec. 2002), 40-447.

[3] Gibson, J.J. *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, MA, 1979.

[4] Helbing, D., Farkas, I., and Vicsek, T. Simulating Dynamical Features of Escape Panic. *Nature*, 407, (2000).

[5] Triandis, H. The Self and Social Behavior in Differing Cultural Contexts. *Psychological Review, 96*, 3 (1989), 506-520.

[6] Woodcock, S. Game AI: The State of the Industry. *Game Developer, 6,* 8 (Nov. 2000).