Game AI R&D Project

Summary Report

Synopsis: This document outlines the final implemented features, technical difficulties encountered and skills gains throughout the Game AI R&D, Immersive AI Project.

Reference: GameAI.Summary v1.0

Date: 23 October 2006

Author: Gavin Bunney & Tom Romano

Status: Definitive

iAI

immersive ai engine

Document Control

Version History

Every change to this document is logged in the table below.

Ver.	Date	Author	Description
v0.1	2006-10-23	Gavin Bunney	Initial document creation
v0.2	2006-10-23	Gavin Bunney	Added immersive AI and development lifecycle
v0.3	2006-10-23	Gavin Bunney	Added final implemented features
v0.4	2006-10-23	Tom Romano	Added skills
v0.5	2006-10-23	Tom Romano	Added time scheduling
v0.6	2006-10-23	Tom Romano	Added conclusion
v0.7	2006-10-23	Gavin Bunney Tom Romano	Added reflection
v0.8	2006-10-23	Gavin Bunney	Extended reflection
v0.9	2006-10-23	Tom Romano	Fixed some grammars/spelling
v1.0	2006-10-23	Gavin Bunney Tom Romano	Definitive Issue

Project Abstract

Implemented game AI, particularly in RPG/MMORPG games, is very script based. The NPC's walk in a set path, speak with set scripts; the mobs have scripted actions. The concept behind the project is to create a more realistic AI, to both provide unpredictability in NPC behaviours, and to immerse a player in the game world.

The project is to research AI techniques, design and implement a goal-based AI system. Goal-based AI is a technique used to create NPC's which act as real players; in that they are given an objective to achieve - e.g. Given a goal of "rake leaves" it is up to the NPC to work out how to achieve their goal, whether through buying a rake, stealing one, or killing another NPC for their rake.

The implemented AI system will be in the form of various classes, created in C++ and Torque Script for the Torque Game Engine, version 1.4. For more information on the Torque Game Engine, visit http://www.garagegames.com.

Contents

List of Figures

1 INTRODUCTION

1.1 Purpose

The purpose of this document is to outline the final implemented features, technical difficulties encountered and skills gained throughout the Game AI R&D, Immersive AI Project.

1.2 Scope

The scope of this document is limited to the Immersive AI engine which was completed on 13th October 2006.

2 IMMERSIVE AI

2.1 Immersive Game World

An immersive game world is an environment which engages players wholly within the confines of the game, through creating an alternate reality for the gamer to explore.

Many key elements make up an immersive game world; graphically intense environments, complex story lines and plots, interactive objects and the reactions of the game world all play a part in the creation of an immersive game environment. With advances in hardware and software, visually immersive environments are now completely achievable. This has lead to further developments and research into game artificial intelligence systems, to help creating game worlds which absorbs the player, making them feel apart of the system.

2.2 Aims & Objectives

The aim of this project was to create a game artificial intelligence system, to assist in the creation of an immersive game world. The implemented system was to allow computer controlled game agents to act, react and interact with the game world in a seemingly intelligent manner.

The key objectives of the implemented system were as follows:

1. To create an artificial intelligence system which controls AI agents that interact with the game as seemingly intelligent entities

2. To design a general system which can be implemented for various goal-states and game types

3. To implement a system which has minimal impact on game performance

3 DEVELOPMENT LIFECYCLE

Throughout the year-long project, 3 main phases of development were carried out. Research and Design were completed in Semester 1, 2006 with Implementation in Semester 2, 2006.

3.1 System Research

The first step in the design of the Immersive AI engine was to research various types of AI systems available and ascertain industry best-practise. This allowed for strong background knowledge of techniques commonly used before any design decisions were made.

It was found throughout the research process that goal-based AI systems were not commonly in use; however have been implemented with success. More commonly AI systems are more tightly controlled to save on computation time and to ensure predictability from the developer's point of view. Testing goal-based systems is much more complex, however in games such as No One Lives Forever 2, it has been seen to both work effectively and create immersive environments for a player.

The research carried out for this project is entitled 'Game AI R&D Project: AI Research'.

3.2 System Design

A design document was created before any work was started on the project. It outlined in detail the various components of the system and how they are to interact with each other. It provided a high-level and detailed design of the system and allowed for the eventual implementation to be performed much more smoothly.

The design decisions made were directly linked to the research carried out earlier and it can be seen that many concepts have been implemented by game designers previously with great success.

The design document for this project is entitled 'Game AI R&D Project: Design Document'.

3.3 Implementation and Testing

Due to the nature of the AI system being developed, the implementation and testing phases were carried out concurrently. Without various components being implemented and tested thoroughly, other components in the system cannot be developed or would not function correctly.

Various C++ and Torque Script classes were developed and once all tested correctly, optimised to the fastest possible implementation. Subversion version control was used to maintain version history and allow concurrent development of parts of the system.

The implementation report for this project is entitled 'Game AI R&D Project: Implementation Report'.

4 FINAL IMPLEMENTED FEATURES

The following features were implemented in the final system; for a more detailed overview of the design, please see 'Game AI R&D Project: Design Document'.

4.1 Goal-based Decision Making

Central to the Immersive AI processing, is the goal-based agent logic. The implemented version utilises a standard 'evaluate' method in each goal/solution, to determine the likelihood of an agent entering the goal/solution.

When an agent requests a goal, all goals available to that agent are evaluated, with the highest being the goal to assign. Similarly the solutions for that goal are then each evaluated and the highest is assigned.

Goals are usually assigned when an agent completed a current goal; however various interrupts can occur to cause the agent to change goals/solutions whilst still completing one. Such things as being attacked can case interrupts, and force the agent to select a new goal based on the new situation. Once the agent completed the interrupted goal, it will return to the previous goal/solution combination that it was executing before it was interrupted.

4.1.1 Technical Difficulties

The main difficulty involved in the goal-based decision making was ensuring that the evaluate methods of each goal are correctly proportional to each other to ensure that agent's make the correct decision based on a situation. Much tweaking of the weighting's and threshold values was done to ensure that the system operated smoothly.

4.2 Goal Lookup Library for Modularity

The Goal Library has been designed to be as modular as possible. There is a central library which holds the goals/solutions available for each agent type. To prevent an agent from assigning a particular goal, it is simply a case of removing that goal from the agent's available list.

The only restriction on each goal file is that it must contain a '<goal>_evaluate' method, which returns between 0.00 and 1.00 to determine the likelihood of an agent assigning that goal.

Similarly the solution files must also contain the evaluate method, but must also implement <solution>_onEnter, <solution>_execute and <solution>_onExit. These three methods are used to form the base for the solution execution state machine; with onEnter setting up the solution, execute performing the actions and onExit cleaning up after the solution.

4.2.1 Technical Difficulties

No technical difficulties were encountered during the implementation of the goal lookup library.

4.3 Different Agent Types

Three different agent types were implemented in the system – Bandit, Entertainer and Solider. These three agent types were designed to represent, in general, the different agent types within a game.

Having different agent types allowed for goals/solution combinations to be altered for specific agents and tested accordingly. The changes of goal/solution combinations force personality changes; such as the Entertainer has a greater change of going into the dance state than the Bandit.

4.3.1 Technical Difficulties

No technical difficulties were encountered during the implementation of the different agent types.

4.4 A* Path Finding

The greatest and most comprehensive part of the system was the path finding implementation. This was due to the lack of previous work on path finding and the technical learning curve needed to develop it within the Torque Game Engine.

The implemented solution works quite effectively. Nodes are placed throughout the game world, interlinked and form a path map for the game world. Currently the path map only generates for the terrain, but can be extended to handle interiors as well (this has been built into the class design).

In order to create a path from one node to another, the A* algorithm was implemented. It allowed for seamless requests for paths from one world point to another by game agents and for them to visit each node and go to the next one in the path.

4.4.1 Technical Difficulties

There were many technical difficulties encountered during the creation of the path finding solution. Initially it was difficult to obtain the terrain topology accurately and to detect when a node is reachable (i.e. not inside an object). Once those problems were sorted out, performance become a big issue; Single path requesting was fine, but if more than twenty agents requested paths, performance dropped dramatically.

Using a profiler, it was found the main performance drop was due to a quick sort being performed on the open node list each time the A* algorithm looped (to sort by node fitness). The solution to the performance was to implement a Binary Heap for the open node list. The Binary Heap allowed much better gain in performance – over one hundred agents can now request paths without performance issues.

4.5 Fight & Flight Reasoning

Fight & Flight reasoning was implemented as part of a goal/solution pair. It contained an evaluate method to determine if the agent was 'strong' enough to fight back or whether to just run away after being attacked.

This implementation allows for a flexible interface for the fight/flight logic – such that the agent type of 'Bandit' always fight, 'Solders' make an informed decision whether to fight, and 'Entertainers' always flight.

4.5.1 Technical Difficulties

Main technical difficulty encountered was to get the agents to direct their fire at the other agents correctly. The issue was caused due to a previous attempt to fix the turning animation of the agents; the fix was changed and the aiming of agents subsequently was corrected also.

4.6 Object Detection and Seeking

The object detection algorithm implemented consisted of a radius search being performed by an agent, every 2seconds, to detect if the object they are after is available. If it was, then the agent created a new path towards that object; otherwise the agent would pick another point to go to and seek the object out.

4.6.1 Technical Difficulties

No technical difficulties were encountered with object detection and seeking as the Torque Game Engine has comprehensive ray cast functions built-in.

5 SKILLS DEVELOPED

When the project started, the Immersive AI team had not previously done any game or artificial intelligence programming. It was an undertaking that was desired to learn about game development, in particular AI systems.

During the research and design phases of the project, much was learned about industry best practise and the way AI systems work in many commercial games. It allowed the design of the system to stem from other's prior knowledge and recommendations and extend it further into a game engine which doesn't have an AI system in place.

The key skills developed include:

- C++ coding – with emphasis on performance of algorithms, memory management
- Using a profiler to find bottlenecks
- Knowledge of Torque Script syntax and development of interfaces between a C++ core and scripting language
- Management of a year long project – handle distribution of workload across a year for two people
- Communication between team members for design and implementation changes

6 TIME SCHEDULING

Throughout the year, the AI system was being developed; whether through the initial research and design phases, through to actual implementation coding/testing. The final developed system consisted of:

- 191 Subversion revision commits
- 9 C++ classes
- 32 Torque Script classes
- 7668 lines of code

The following is the timetable developed for the project, illustrating the various milestones and distribution of work for each component of the system:

	Task Name	Duration	Start	Finish
1	**Research Game AI**	**10 days**	**Mon 27/03/06**	**Fri 7/04/06**
2	Industry Best Practise	5 days	Mon 27/03/06	Fri 31/03/06
3	Limitations	5 days	Mon 27/03/06	Fri 31/03/06
4	Goal-Based Techniques	5 days	Mon 3/04/06	Fri 7/04/06
5	Bayesian Networks	5 days	Mon 3/04/06	Fri 7/04/06
6	A* Path Finding	5 days	Mon 3/04/06	Fri 7/04/06
7	**Design Goal Modules**	**10 days**	**Mon 10/04/06**	**Fri 21/04/06**
8	Design Seek	10 days	Mon 10/04/06	Fri 21/04/06
9	Design Attack	10 days	Mon 10/04/06	Fri 21/04/06
10	Design Interaction	10 days	Mon 10/04/06	Fri 21/04/06
11	**Game Prototype**	**30 days**	**Mon 24/04/06**	**Fri 2/06/06**
12	Core Game Engine	15 days	Mon 24/04/06	Fri 12/05/06
13	A* Path Finding & Object Recogr	15 days	Mon 15/05/06	Fri 2/06/06
14	**Class Creation/Implementation**	**40 days**	**Mon 10/07/06**	**Fri 1/09/06**
15	Core Engine Changes	10 days	Mon 10/07/06	Fri 21/07/06
16	Seek Class	15 days	Mon 24/07/06	Fri 11/08/06
17	Attack Class	15 days	Mon 24/07/06	Fri 11/08/06
18	Interaction Class	15 days	Mon 14/08/06	Fri 1/09/06
19	**Product Finalisation**	**10 days**	**Mon 4/09/06**	**Fri 15/09/06**
20	Testing & Changes	10 days	Mon 4/09/06	Fri 15/09/06
21	**Deliverables**	**145 days**	**Fri 7/04/06**	**Tue 24/10/06**
22	Research Report	0 days	Fri 7/04/06	Fri 7/04/06
23	Design Document	0 days	Fri 5/05/06	Fri 5/05/06
24	Demonstrate Prototype	0 days	Fri 9/06/06	Fri 9/06/06
25	Implemented Solution	0 days	Fri 13/10/06	Fri 13/10/06
26	Summary Report	0 days	Tue 24/10/06	Tue 24/10/06
27	Poster Presentation: 10%	0 days	Tue 24/10/06	Tue 24/10/06
28	Oral Presentation: 10%	0 days	Tue 24/10/06	Tue 24/10/06

Figure 6-1 Project Timetable

7 REFLECTION

The Immersive AI system, from initial conception, through the final implementation has been an excellent learning project for the Immersive AI team.

The initial concept was to create an AI system with similar capabilities as the one implemented in *Elder Scrolls: Oblivion*, due to the impressive AI capabilities it displayed. Through the initial research conducted, a more concise project aim and objectives were devised, as a more comprehensive knowledge of AI systems was obtained.

From this background knowledge a design of the AI system was developed. The original design was still used in the implementation of the system, so it can be seen that the initial design considerations were complete and extensive.

The component of the system which took the most time would have been the path finding. As this was the first and most complex part of the system, much had to be learned both about how to develop the path finding system and how to use the Torque Game Engine to develop it. Looking back it would have been more effective to develop the goal/solution system first, to develop knowledge of Torque, and then complete the path finding. However, with the knowledge gained through the various attempts at creating a path finding system, the other components of the system were much easier to understand and implement.

The core functionality of the system was the development of the goal/solution manager; initially a list of goals and solutions to implement was devised. This list was integrated into an excel workbook, to allow for a visual representation of the events to cause an agent to select the goal or solution. Towards the end of the implementation of the system, it was found that many of the weights and properties used to describe the probabilities of an agent selecting a particular goal or solution were insufficient to cause the agent's to change their state. This was corrected during the final testing phase and was simply a case of changing the evaluate methods of each of the goals and solutions.

Throughout the project, the use of subversion has been instrumental in allowing the simultaneous development of the AI system and without it would be almost impossible to manage the changes made.

Overall, the Immersive AI team believe the system is comprehensive, modular and complete implementation of a goal-based AI system. Within the engine is a robust path finding system and an object detection/recognition system. The developed engine components are flexible enough to be integrated into another game or simply extended for further functionality.

8 CONCLUSION

During the initial development, the system was designed around industry best practise and what had been implemented previously in some games. The eventual implementation was consistent to the original design and works quite effectively. Around 70 agents can be active in the game world without any performance change, which well exceeds initial expectations of around 40 agents.

Many technical difficulties associated with the development of an artificial intelligence system were overcome throughout the life of the project, through learning new techniques in both C++ and Torque Script.

The Immersive AI system, from initial conception, through the final implementation has been an excellent learning project for the Immersive AI team. The final Immersive AI system meets all initial project expectations and can be utilised inside a complete game for a realistic, immersive, artificial intelligence system.

9 GLOSSARY

Item	Description
AI	Artificial Intelligence - The ability of a computer or other machine to perform those activities that are normally thought to require intelligence
Agent	A computer controlled entity within a game environment
Goal	The purpose toward which an endeavor is directed, an objective; describes the AI agent's current overall objective.
RPG	Role-Playing Game - A game in which players assume the roles of characters and act out fantastical adventures, the outcomes of which are partially determined by chance
Solution	The resolution to achieve a goal
State	A condition of being in a stage or form; describes the AI agent's current action algorithmic state

10 REFERENCES

- Bunney, Gavin & Romano, Tom. “iAI Research” 2006 irombu.com, Brisbane.
- Bunney, Gavin & Romano, Tom. “iAI Design Document” 2006 irombu.com, Brisbane.
- Bunney, Gavin & Romano, Tom. “iAI Implementation Report” 2006 irombu.com, Brisbane.