Title

<Full Name>

Submitted for the degree of Msc Artificial Intelligence

Heriot-Watt University

<School/PGI>

<Month> <Year>

Acknowledgements

Abstract

Table of Contents

[1. Introduction 1](#_Toc350176725)

[2. Objectives 2](#_Toc350176726)

[3. Literature Review 3](#_Toc350176727)

[3.1 Game Engines 3](#_Toc350176728)

[3.1.1 Unreal Engine 3](#_Toc350176729)

[3.1.2 Cry-Engine 4](#_Toc350176730)

[3.1.3 Unity3D 4](#_Toc350176731)

[3.1.4 Blender 4](#_Toc350176732)

[3.1.5 Game Maker 4](#_Toc350176733)

[3.1.6 Loom 4](#_Toc350176734)

[3.1.7 Overall Choice of Game Engine 4](#_Toc350176735)

[3.2 Current Game Standards 5](#_Toc350176736)

[3.3 Evolutionary Games 6](#_Toc350176737)

[3.3.1 Galactic Arms Race 6](#_Toc350176738)

[3.3.2 Nero 6](#_Toc350176739)

[3.4 Neural Networks 8](#_Toc350176740)

[3.5 Interfacing In-between Games 9](#_Toc350176741)

[4. Methodology 10](#_Toc350176742)

[5. Requirements 11](#_Toc350176743)

[5.1 Aims 11](#_Toc350176744)

[5.2 Objectives 12](#_Toc350176745)

[5.3 Requirements 13](#_Toc350176746)

[6. Professional, Legal and Ethical Issues 14](#_Toc350176747)

[7. Project Plan 15](#_Toc350176748)

[7.1 Gantt Chart 15](#_Toc350176749)

[7.2 Stages of This Project 15](#_Toc350176750)

[7.2.1 Setting up the project 15](#_Toc350176751)

[7.2.2 Research Report 15](#_Toc350176752)

[7.2.3 Creating the Game 15](#_Toc350176753)

[7.2.4 Creating Interface 16](#_Toc350176754)

[7.2.5 Create Neural Network 16](#_Toc350176755)

[7.2.6 Experiments 16](#_Toc350176756)

[Bibliography 18](#_Toc350176757)

Table of Figures

[Figure 1 – A Gantt chart showing the timeline for this project. 11](#_Toc349311195)

# Introduction

# Objectives

The main objective of this project is to create an interface to sit between a game and an external application. The external application will control a specific part of the game. This project aims to edit the behaviour of a character within the game. The game engine/game will output data to the interface which will in turn pass it to the external application. The external application will then pass back new data on what the in game character should do.

This project aims to use a neural network to train the character in game to be able to solve simple puzzles. These could include move a box off a switch, press a button to open a door.

The interface will be a layer that will sit between the game engine and the external application, whatever that may be. The interface should be flexible and allow for general data to be passed between its outputs.

The idea for the interface is that it should be able to handle multiple game engines. This gives developers the ability to reuse software that they have already written.

For example if a developer has written a controller for an AI in a racing game. Instead of re-writing it for every game engine that the need it for they use the interface as a medium between. The developer will have to go into the game and hook up all the proper connections but after that they can swap out the controller for another.

This gives the developer the ability to re-use software and it also makes the process more modular. The behaviours can be swapped out like Lego bricks.

# Literature Review

## Game Engines

This project will require a game engine. A game engine is a tool that allows for developers to create games on. Think of it like a framework that contains all the tools that a game developer would generally need.

With a wide number of game engines available for use in this project, there needs to be criteria to select the game engine that will be the most suitable for this project.

The first piece of criteria will be that the game engine is free to use. This project requires the game engine be free to use, wither that being an open source game engine or a professional engine that is free to use for academic use.

The next piece of criteria is that it is quick to learn. Due to the scope of this project and the time limit available, the author feels that in choosing a game engine that will take 6 months to learn how to code for is not applicable for this project. Therefore the game engine must be straightforward to develop for.

Next is the level of access available to the developers to the game engine. In order for an interface to sit between the game engine and an external application the developer will need access to some of the lower level functionality of the game engine. This could include things like networking features, restricting certain override functions. This will be needed when it comes to synchronising between the interface and the game engine.

Criteria of the game engine that this project is not concerned with are features like if the game engine is 2D or 3D, the overall look of the end game (graphics), sound capabilities and release platforms. These features are not exactly needed for this project therefore they should not be taken into consideration when deciding upon a game engine.

Based on these criteria the following game engines have been selected:

### Unreal Engine

This is one of the oldest game engines on the list. Currently on its third version, fourth is about to be released at time of writing. This is a professional game engine that a lot of industry game developers use for AAA titles. Such games include Batman: Arkham series (2009) and the BioShock series (2007) were created in this engine. This engine is free to use for non-commercial use, meaning that it can be used for free in this project. This engine uses its own scripting language called UnrealScript. While this engine is one of the industry standards, the fact that it uses its own language that the author will have to learn as well as the engine, makes this engine an unlikely choice due to time constraints.

### Cry-Engine

The cry-engine is one the current pinnacles of game engines. This engine was developed by Crytek and has been featured in many AAA titles, such as the Crysis series. This game engine has scripting in LUA and has C++ in the game engine. While these are both great languages, which are used in professional game development, the time it will take to learn not just the engine but the languages as well. The cry-engine is also free to use, for none commercial use. Since this project will not be released then this fully complies with their licensing

### Unity3D

Unity3D is a game engine that has been recently became a wide hit with the indie game development community. This is due to its ease of programming for and the fact that it is free to use. There are two versions of this game engine, free and pro. The pro version allows developers to use the more advanced features and removes watermarks. The game engine is a full professional game engine; it was created by professionals, not just an open source game engine that a group of people have hacked together. Along with the pro version, developers can buy licences for certain platforms such as Android, Xbox 360 and PlayStation 3 to name a few. As for languages the game engine supports three natively. These are C#, JavaScript and Boo (language based on python). All three of these languages are relatively simple to develop in.

### Blender

Blender is an open source 3D modelling tool that has a game engine built in. Since it is open source then that means that this meets the free to use criteria. Also it allows the developer to access the lower features of the game engine. It is written in python, which is a relatively simple language compared to other game engines, such as Unreal engine which is in UnrealScript.

With above features it makes it a strong contender for this project.

### Overall Choice of Game Engine

Based upon the requirement criteria given above the selected game engine will be Unity3D. This engine was chosen as it met all the requirements stated above. It is free to use, its ease of learning and its strong documentation, both professional and community of developers.

## Current Game Standards

At the present there are two parties in artificial intelligence, game developers and academic researchers. While the academic researchers have far more advanced techniques than the games industry, the games industry is something to accept these new techniques. Currently games industry uses pathfinding and steering behaviours, and that is about it. More advance techniques are not used, such as bio inspired techniques. This is due to a number of reasons, mainly due to developers focus. In the games industry there is one main focus, graphics. Graphics in a game is right now, and for at least the last 10 years, is the most focused on part. So much so that current generation consoles have multiple cores just for graphics and only a single dedicated to everything else.

Another reason is processor constraints as mentioned in Current AI in Games: A review (Sweetser and Wiles 2002). This paper goes on to mention the drawbacks of using more advanced AI techniques within games. The paper states that game developers are reluctant to produce games that have learning techniques, such as neural networks and genetic algorithms, in case they develop/learn stupid behaviours. Also in the case of genetic algorithms they are very computationally expensive, something the game cannot have due to the amount of other tasks that need to be carried out.

While most modern games only take advantage of steering behaviours, state machines and A\*, there have been a few commercial games that have been released with more advanced AI techniques.

These games include the Black & White series (2001) which feature the player praising or punishing the in game character based on the actions. For example if the creature attacks someone then you can punish it, therefore it knows that attacking people is wrong.

(Yannakakis 2012)

(Alexander 2004)

(Millington and Funge 2009)

(Buckland 2002){Yannakakis, 2012, Game AI Revisited}{Yannakakis, 2012, Game AI Revisited}{Sweetser, 2002, Current AI in Games: A review}{Sweetser, 2002, Current AI in Games: A review}

{Sweetser, 2002, Current AI in Games: A review}

## Evolutionary Games

As discussed above, current game developers are reluctant to use more advanced artificial intelligence techniques in their games. Although some academic researchers have tried to prove that these techniques can be used within games.

While most of these do not go on sale, they instead become freeware, they are still games.

### Galactic Arms Race

(Hastings et al. 2009)

Galactic Arms Race was a game made by students at the university of

The game features a method of generating content within the game at real time. The content generated is based on the player preference.

The game uses neural networks to evolve weapons based on which weapons the player uses the most. Each weapon is made up of a neural network, at every frame of the weapons animation parameters are passed through it and the particles moved according to the output of the neural network.

This proved that not only can content be generated within a game but it can be also generated to a players preference.

### Nero

This is a project by a group of academic researchers.

This was aimed to create behaviours at real time that would evolve to solve a problem. For example if the player wanted their in game characters to shoot the enemy then a genetic algorithm is used to evolve the neural network that controls the character.

It uses the NEAT method of altering the neural network.

(Stanley et al. 2005)

## Neural Networks

This project will feature a neural network to evolve the behaviour of a character in game.

## Interfacing In-between Games

# Methodology

This section will detail how the author will develop the project.

## Prototypes

### Prototype 1

Prototype 1 will feature the robot in the game moving around the environment with a simple wander behaviour. There will be no interface between the game engine and the wander behaviour. The behaviour will be coded into the game.

### Prototype 2

Prototype 2 will feature the game sending messages from the game to the interface. This will be based upon what object it can see when it fires a raycast. The interface will print out the objects name to the console.

### Prototype 3

Prototype 3 will feature the same wander behaviour but being fed through the interface, rather than being hand coded in.

### Prototype 4

Prototype 4 will feature the neural network instead of the wander behaviour. This will be fed data from the interface.

### Prototype 5

Prototype 5 will feature the neural network leaning to solve simple puzzles within the game, these could include simple stand next to door to get it to open, stand on switch to finish level.

# Requirements

## Risk Assessment

With this project, and any project, has the potential to fail. This chapter aims to point out the main points at where this project is likely to fail.

### Transferring data

This project has one key failure point and that is when it comes to transferring data to the interface. This one of the key points of this project and if this point fails then so does the whole interface part of this project.

## Performance Assessment

## Requirements List

The tools required for this project are:

A repository

A repository hosting tool

A project management tool

A game engine

An IDE for developing the interface and the neural network

### Game Engine

As stated above in 3 Literature Review this project will require a game engine to create the game in. Therefore the author will need to install the game engine and all the necessary tools that accompany it.

### Project Management

This project will require a method of managing the project. This involves showing all the tasks that need to be finished as well as how long they should take, etc. For this project a tool called pivotal tracker will be used. It allows the user to manage tasks effectively as well as provides a document at the end showing the statistics of the project. This includes how many tasks, how long it took, any problems with it etc.

### Version Control

Since this is a software project not having version control would not be advisable. The two main choices when it comes to version control are SVN and GIT. Both provide the same service, it just comes down to user preference. Since the author has previous experience with GIT, this project will use it.

Hosting the GIT repository is another user preference. This project will use Github, as it provides graphs and charts to show commits and it also has its own wiki for each repository. These two features aren’t a killer feature but they are nice.

### Development Tool

The neural network and the interface need to be developed in an IDE. This tool will be dependent upon what language is needed for the game engine. Since the Unity game engine is used then the language will be C#. Therefore the IDE tool will be visual studio. Either visual studio 2010 or visual studio 2012 will be used, as the author has access to both.

# Professional, Legal and Ethical Issues

## Professional Issues

This project will not break any of the BCS codes of conduct. Therefore the author will not be breaking any professional issues.

## Legal Issues

This project does not come across any legal issues. The only legal issues that it may come across are if the user uses it without a licence for the game engine selected in the literature review.

## Ethical Issues

This project will not come across any ethical issues. It cannot be used for un-ethical reasons.

There will be no human test subjects or even human testers. This will be fully human free with the computer doing all of the experiments.

# Project Plan

## Gantt Chart

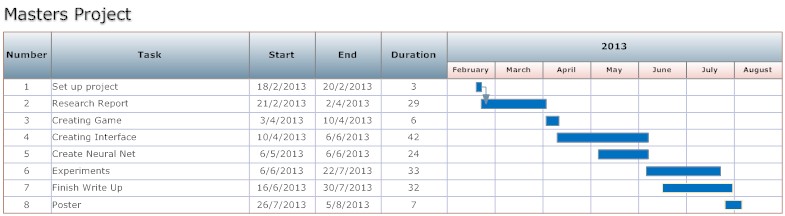


Figure 1 – A Gantt chart showing the timeline for this project.

## Stages of This Project

As stated in the above Gantt chart [Figure 1] this project has various stages. These stages are:

### Setting up the project

This project requires setting up before starting in order to

Setting up a repository is one of the most important tools that will be used in this project. Either GIT or SVN can be used, but for this project GIT will be used. This requires little in the way of setting up. This repository will store all the files needed.

Next pivotal tracker will be set up to manage deadline and tasks that need finished. Pivotal tracker is used in agile software development but it can still apply to this.

Lastly is formatting the research report correctly. This involves setting up the report into the correct academic format.

All of this should take a week to do.

### Research Report

Writing the research report will take up a substantial amount of time up until the third semester. That is why no other project work will take place until this is finished.

### Creating the Game

The game is a key part to this project. The game will not be fully polished like main games. It will just serve as a platform to feed the data to the interface. This will take place immediately after the research report is finished. Once this is finished the game will not involve major work, maybe some tweaks when it comes to optimisation communication between this and the interface.

### Creating Interface

The interface is the key part of this project and therefore will require the most time. Linking this to the game will take a substantial amount of time.

### Create Neural Network

The neural network will not require a substantial amount of work. The main reason this section will take so long is because it will require a lot of back and forth work between this and the interface.

### Experiments

Lastly once all the implementation is finished testing will take place. This will involve testing the neural network in the game to prove that it is learning and behaving in a sensible manor.

Bibliography

Alexander, N. (2004) 'AI in Computer Games', *Queue,* 1(10), 58-65.

Buckland, M. (2002) *AI techniques for game programming,* Course Technology.

Games, I. and Marin, L. (2007) BioShock, email to [accessed

Hastings, E. J., Guha, R. K. and Stanley, K. O. (2009) *Evolving content in the galactic arms race video game,* translated by IEEE, 241-248.

Millington, I. and Funge, J. (2009) *Artificial intelligence for games,* Morgan Kaufmann.

Stanley, K. O., Bryant, B. D. and Miikkulainen, R. (2005) 'Evolving neural network agents in the NERO video game', *Proceedings of the IEEE*, 182-189.

Studios, L. (2001) 'Black & White',

Studios, R. (2009) Batman: Arkham Asylum, email to [accessed

Sweetser, P. and Wiles, J. (2002) 'Current AI in Games: A review', *Australian Journal of Intelligent Information Processing Systems,* 8(1), 24-42.

Yannakakis, G. N. (2012) 'Game AI Revisited'.