The A.I Craft Project

Ray Shulang Lei
200253624
ray.lei@uregina.ca

January 14, 2012

Abstract

This is the proposed paper of a multiplayer cloud base livecoding A.I combating game, similar to robocode.

1 Introduction

A.I craft is an online A.I combating game. Players use their livecoding skills to provide code in real time to control their A.I crafts, in order to beat the opposite players. The idea is inspired by the JAVA Robocode[1] project.

2 Reasonale

In Noah Falstein's article "Natural Funativity"[2], he described three types of hunters in a hunter-gatherer society. After three hunters have got enough haunch of antelope, enough to feed his family for a few days:
Aagh - goes right back out to track down a deer he saw earlier;
Bohg - passes his time by kicking back and catching nothing more challenging than some rays;
Cragh - plays hunting games at home. "Like Aagh, he is building survival skills - but does so it in the safe confines of home. Like Bohg, he stays safe - but also stays fit and slightly improves his chances for success in the next

hunt. ”
Noah Falstein concluded that ”Over hundreds of thousands of generations, those genes that Cragh carries are more likely to spread, and the activities - including games - can also be passed on through word of mouth in the tribe from generation to generation.”

According to Venkatesh’s article ”The Rise of Developeronomics” [3] on Forbes, our society now is a developer-centric economy. Thus there are only two kinds of people matters: developers, and people who controls developers. Assuming our society is developer driven, I can conclude that the developers are as important as hunters in a hunter-gatherer society. What Aagh, Bohg and Cragh would be like in today’s society? After developers finishes their work before the deadline, there would be three types of behaviors:
Aagh - goes right back out to code for more income, whether it is a contract job or freelancing;
Bohg - will be chilling out, get more sleep, let the brain rest;
Cragh - plays games at home to sharpen his development skills.

Now, let us wait a second and rethink about what kind of games Cragh can use to improve his development skills. First person shooters? It only makes a better hunter. Strategy games and others? Maybe, but they are still not exactly related to coding. Thus we need a new genre of games: coding games. If Noah and Venkatesh are right, developers who plays coding games are more likely to spread their genes. And right here I have found the goal of making A.I Craft: to imporve the chance of spreading my genes. Therefore it makes perfect sense to put my best effort fulfilling it.

3 Core Design

The basic idea is very similar to Robocode. Players provide A.I controlling code to tell their crafts what to do. The game engine provides 4 sets of controlling APIs such as move, search, aim and fire. To satisfy my goal of improving Cragh’s ability to code while making the process fun and fair, I have to make the controlling A.I fair to some extent. That’s why players have to use livecoding skills. According to wikipedia, livecoding can also be referred to as improvised interactive programming. My players will have to start from scratch, and improvised their A.I from there. And I will have to provide some mechanism to prevent copy and paste.

4 Technical Details(Draft)

4.1 NodeJS

The server side will be back by NodeJS. Since users will be submitting javascript code in real time to control their crafts, choosing NodeJS seems obvious. NodeJS will be used to fulfill these features:

1. Serving front end HTML/WebGL files
2. Connecting to database
3. Checking if user submitted code is secure
4. Executing user submitted code in real time
5. Executing and storing game logic in real time

4.2 WebGL/Javascript in Browser

The client side part of A.I Craft will be back by Html5/WebGL in your browser. The animation and physical calculation of the game will be in this part. Features included:

1. Rendering the game scene, player crafts and other game objects
2. Calculating collision detections
3. Submitting players codes in real time
4. Preventing players from submitting pre-coded segments

4.3 Database engine

The databases will be seperated as client and server sides. SQL is my first choice due to pass experience. NoSQL might be considered with further research. Basic idea is that game objects information stores at client side databases, and they sync with server side database once a while in a timely basis according to network performance.

References

[1] Mathew Nelson, Flemming Larsen, IBM etc.
Robocode.
http://robocode.sourceforge.net
2001

[2] Noah Falstein,
Natural Funativity.
http://www.gamasutra.com/view/feature/2160/natural_funativity.php
2004.

[3] Venkatesh Rao,
The Rise of Developeronomics.
http://www.forbes.com/sites/venkateshrao/2011/12/05/the-rise-of-developeronomics/
2011.