**Software Maintenance Document for Master’s Project**

Chen Gao, MIT

Flinders University

School of Computer Science, Engineering, and Mathematics in the Faculty of Science and Engineering

Aim

The aim of this document is providing an overview of my coding to successors who may keep exploring player behaviour prediction method in future.

Constrains

All information contained in this document is based on project files submitted to git-hub at 24 Aug 2015, reference number 33f5d5daf81c3c0186bf70ee3f61404c6b72a68, code can be browsed at following address:

https://github.com/maicallist/MasterThesisProject/commit/33f5d5daf81c3c0186bf70ee3f61404c6b72a689

Setup Environment

Project was built based on Microsoft XNA 4.0 using Microsoft Visual Studio 2013. Install VS2013 XNA 4.0 Refresh plugin, if program indicates error, try installing XNA 4.0 (not 4.0 refresh) again, installation may fail, but then program should work.

Code Overview

The entire game framework is borrowed from one of my previous game. It handles player input (InputManager class), how game screens transit (ScreenManager), animation effect (FadeAnimation) and so on. All master project related codes are in two class: DataCollector (single player mode) and GamePlayScreen (multiplayer mode).

DataCollector Class

In single player mode, we try to collect various data to create prediction models. Players have following controls:

Left arrow key and right arrow key – move paddle at bottom of screen

Space – start or restart game

XNA framework keeps tracking elapsed game time, therefore, in update() line 223 (@223), we updates current time tag every time we call update(), if in current cycle of update, we successfully collect some data, store them with current time tag.

Recording Prediction Data

In update(), we call doPrediction() to check if current game state fits our prediction model. @1020-1083,

GamePlayScreen Class