SIT352 Advanced Game Development

Assignment 1 - ShroomBot!

Due Dates:

First bot submission - due no later than 5:00 PM, Friday 8th April

Final bot submission - due no later than 5:00 PM, Friday 29th April

Report submission - due no later than 11:59 PM, Sunday 1st May.

Introduction

This assignment considers two important aspects of game AI: making decisions under resource constraints with limited information; and, being able to select between potential solutions based on their performance on a specified task. This task will require you to develop and submit for performance evaluation at least one bot capable of playing the game "Shroom", as well as a brief report on your bot development over the course of the task. Assessment of your submission will take into account:

- Development report (**30 marks**)
  - The methodology/algorithms employed to play the game autonomously;
  - Your code implementation of the bot(s);
  - A brief description of how you improved upon existing algorithms (yours or those of your fellow competitors) to improve bot performance.
  - A brief discussion of the aspects of the game AI problem that you found difficult and how you overcame them (or failed to)

- Bot Performance (**70 marks**) – described below

To satisfactorily complete this task you must accrue 50 marks or greater. Information on how marks are awarded for Bot Performance is given below. ***In addition to formal assessment, there will also be a prize for the best performing bot submitted over the course of the assessment task.***

The Shroom Game

Figure 1 shows a screen shot of the game Shroom, which has been developed specifically for this assessment task. The aim of the game is simply to find and eat the Magic Shroom (♣). The ShroomBot, written by the student, controls the *Hunter* character (☺). Throughout the game world are scattered Shrooms (♠). The Hunter can consume these as well. Some Shrooms will aid the Hunter by providing energy; others will harm the Hunter by reducing its energy. To find the Magic Shroom, the ShroomBot must decide at each turn of the game, whether to consume available Shrooms or not, which direction to turn or to step forward. The Hunter cannot walk through walls or Shrooms. Unfortunately the game world is shrouded in mist, which clears as the Hunter passes through it, making it impossible for the Hunter to travel directly to the Magic Shroom.

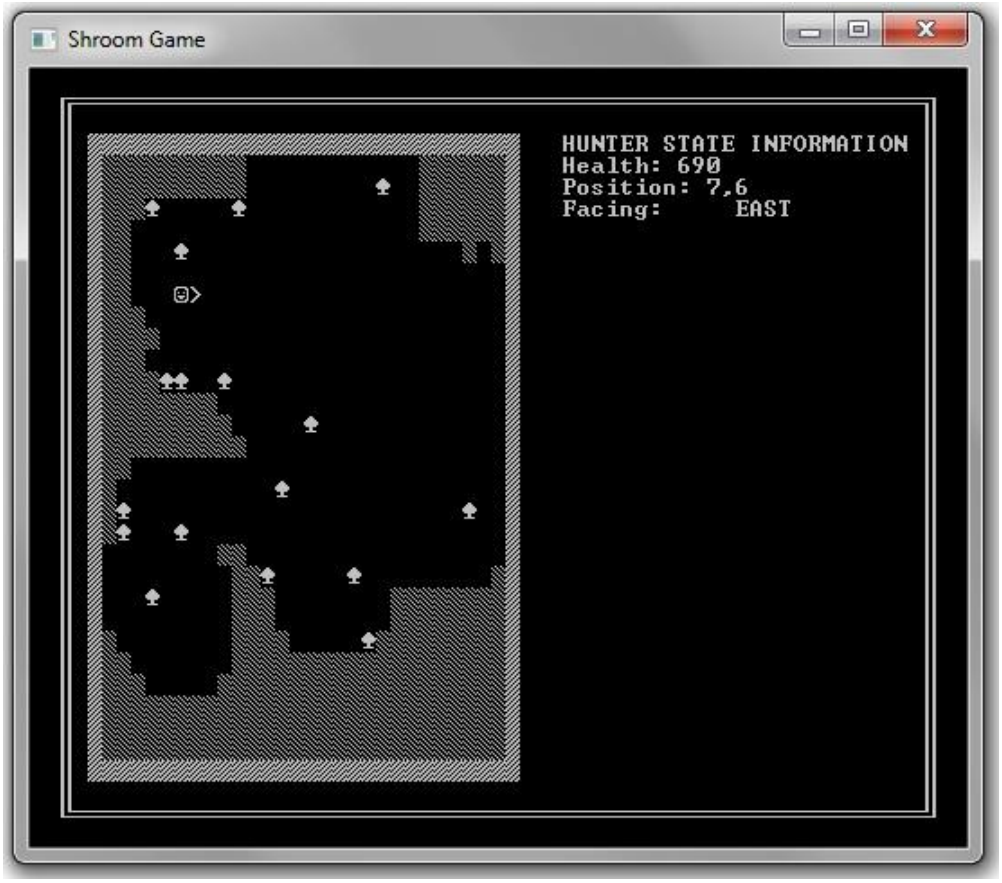

Figure 1: Screenshot of Shroom game

As the Hunter can only eat Shrooms that are directly in front of it, and the location of the Magic Shroom is initially unknown, the Hunter must explore ShroomWorld in order to find it. Unfortunately, the Hunter has only a limited amount of energy, which is consumed when it moves in the world. In most cases the Hunter will not have sufficient energy to explore the entire world to find the Magic Shroom without consuming Shrooms. Exploration (particularly efficient exploration) and intelligent use of the Shrooms as a resource, is required to perform well in this game.

Programming the ShroomBot

All students are provided with a Visual Studio project for the ShroomBot DLL. Essential elements of the application - necessary to permit the Shroom game to load, communicate with and unload the ShroomBot - have been provided. These elements of the code should not be modified in any way. Students must provide a class, within the DLL, derived from the given IController interface class. Specific details on how to integrate their class into the existing ShroomBot DLL application are provided in the supporting documentation for the ShroomBot project.

Each turn in the game the ShroomBot is activated via its OnTick method, which also passes a percept regarding the Hunter's local state (its health and what it can see in the surrounding cells of the game world). The Hunter must respond with an action, by calling one of the appropriate methods of the IControllable interface (ProcessAction OR QueueAction and ProcessQueue), exposed by the Hunter object that the bot is controlling. Both interface specifications are provided in the ShroomBot.h header file. The file MsgTypes.h gives information about the data types relevant to the Bot. A demo bot called BlindBot is provided to show students how to achieve basic movement within the game world. More information regarding the programming of a ShroomBot is provided in the Shroom Documentation on DSO.

Performance Evaluation

Performance evaluation will consist of five (5) rounds (iterations) of the following weekly schedule. Students will have until 5:00 PM on the Friday of each week to submit a bot for that round. Students MUST submit a first attempt at a bot within the first two rounds (either for round 1 evaluation or round 2). This bot does NOT have to achieve performance standards appropriate to a pass grade (although students should try to get a basic good attempt going). Failure to submit a bot before the closure of round two submissions will result in a 50% mark penalty for this assessment.

Game performance evaluation will be conducted at the closure of each round and results posted to DSO by 9:00 AM on the following Monday. Students may enter as many rounds as they desire and they may submit more than one bot per round. However, each bot submitted in a given round MUST implement a significantly different algorithm to be counted in the standings for that round. Where the judges determine that two (or more) submissions from the same entrant are essentially the same algorithm, one will be chosen randomly for assessment.

Game Performance score will be based on percentage of games one over N trials (where N will be of the order of 100). The discriminator, should two bots achieve the same performance, will be efficiency (the bot with the least deficit of energy, normalised against

the starting energy). Where there are multiple tasks for the round, the final performance score will be the average of the scores obtained on the individual tasks.

The table below identifies the information for each round, including the Task for that round and the means by which points may be earned during that round. The top 3 bots in a given round will automatically be entered into the following round. This has three significant implications: (1) that an entrant who earns a top 3 spot in a round can only enter a bot into the next round if it implements a significantly different algorithm (this encourages diversity); (2) that to achieve a top bot position, you must knock off the reigning champion and beat everyone else in that round (the bar gets raised as the competition evolves); and, (3) if you can write a good bot early and it remains good, you'll do well!

Round #	Submission Deadline: 5:00 PM Friday on	Task	Points Categories
1	1st April	Map Reveal	A,B
2	8th April	Map 1	A,B,C
3	15th April	Maps 1 & 2	A,B,C,D,E
4	22th April	Maps 1, 2 & 3	A,B,C,D,E
5	29th April	Maps 1, 2 & 3	A,B,C,D,E

Points Categories:

A. Achieve the top game performance score during that round: 50 points
B. Achieve a Top 5 place (not 1st) in game performance score during that round: 30 points
C. First submission of a given algorithm: 150 points
D. Achieve at least 80% of the game performance score of the previous week's top bot: 150 points
E. Surpass (by >5%) the game performance score of the top bot of the previous round: 50 points

To achieve a pass grade in this assignment, students will need to accrue a MINIMUM of 300 points from Bot assessment. This translates into obtaining 20/70 marks from this portion of the assessment; to obtain a pass grade in this circumstance the student would need to obtain 30/30 from the report and code evaluation part of the assessment. Clearly students should attempt to accrue as many points as possible in bot evaluation. A score of 1000points will earn 70/70 for the Bot assessment task.

Tasks

Specifications for each task, including details of the configuration file to be used, will be provided in supplementary documentation on DSO.

Submitting a Bot for Evaluation

To submit a bot you must provide your complete Visual Studio ShroomBot project containing all code relevant to compiling the DLL. You MUST clean your project before placing it in a ZIP archive (to clean, select Build->Clean Solution) menu option.

Bots that do not compile, or cause run time errors, will not be evaluated and will score 0 points.

You MUST also provide a brief (half to one page) description of the algorithm that your bot implements. This description will be made public if your bot achieves a top 5 place during that round. Your description must contain sufficient detail to allow:

- reimplementation of the method by another person (but does not need to include code or coding explanation); and,
- evaluation of the novelty of the algorithm by the judges.

The point here is to share ideas with your peers and then try to beat them at their own game in the competition!

Final Report

Your final report should consist of:

- The brief algorithm descriptions provided when you submitted your bot, including any modifications you made if the algorithm was originally posed by another student;
- A one page (minimum) discussion of the aspects of the problem that you found most challenging to overcome. You should relate these problems to the specific algorithms that you implemented (either in describing how you overcame them, or how your algorithms fail to overcome them).
- As an appendix, all code written by you to implement your algorithms. Your code must contain clear comments and be neatly and logically organised.

Your final report is due by 11:59 PM, Sunday 1st May.