CSC258 Project Proposal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | First Name | Last Name | Student No. | Email |
| Member 1 | Ming Feng | Wan | 1002531236 | Ming.wan@mail.utoronto.ca |
| Member 2 |  |  |  |  |

What is the title of your project?

Learning Rock Paper Scissors

Provide a one-paragraph description of your project.

Possible methods for the Rock Paper Scissor AI:

1. Completely random
2. Markov model (<http://setosa.io/ev/markov-chains/>)
3. REINFORCE (if we have time, might need to use memory) (<https://www.cs.toronto.edu/~guerzhoy/411_2018/lec/week10/REINFORCE.pdf>)
4. q-learning (if we have more time)
5. AI that predicts move by human psychology (Backup plan if Matrix computation encountered problems)

Project Description:

(This is where you describe your project in detail. You can use the Design Case Studies slides as a reference on how to create the following components for your project. All of these components are not compulsory for you to have but most projects usually have these components. They are: high level pseudo code, state diagram, datapath-and-control-CU block diagram, input/output block diagram [Example: <https://www.nandland.com/goboard/images/project10-pong-block-diagram.png>], truth tables etc. These components should be designed and described to show your understanding of your design i.e. how many bits is each input/output, what is the max number your counter can count up to, how many counters/shift registers you need to use etc. A good idea is to get an initial draft of this done and show to your TA in the next lab or during office hours to get feedback.)

Front end rock paper scissor interface design

Operations required for Backend computation

* Float numbers
* Matrices (2D or 3D array)
* Multiplication
* For loops
* Random numbers (<https://en.wikipedia.org/wiki/Linear-feedback_shift_register>)

Timed game

Reason: everyone plays rock paper scissor with a pattern; An AI can “learn” the pattern and beat the player

Might be added: Pseudo code for Markov and REINFORCE

Components: VGA/displayer for the game; FSM for Rock Paper Scissor display and to start/end game, method selection

What will you accomplish for the first milestone?

(Advice here: Pretend that you're designing Lab 8 around your project idea, in keeping with the difficulty level of the previous labs. Try to be **specific and detailed** in describing the components that you will complete. Don't say that you'll "think about" or "plan" or "design" something.

Bad example 1: We build the graphical interface.

Good example 1: We display moving spaceships on VGA.

Bad example 2: We write code for the PS2 keyboard interface.

Good example 2: We make the PS2 keyboard work and show the key inputs on the HEX display.

Assume your project can be developed in three independent parts, what you write in the space below should outline the components of the first part. Make sure to describe a full lab's worth of work, including the evidence of your work that you will provide to the TAs to justify getting the full marks for this milestone.)

Display the gaming interface with FPGA keys as rock, paper, scissor, and reset, score calculator, and timer

Finish the Markov model

What will you accomplish for the second milestone?

(similar advice as above, but for the second part of your project. Remember to specify what inputs and outputs will be used for each milestone. If your project is a visual game for example, what will appear on the screen for each milestone, for example, static colored boxes in one milestone and moving boxes in the next one etc.)

Fully test the project

Polish the front end

What will you accomplish for the third milestone?

(don't say "everything" just because this is the final milestone; describe the final components instead, and exactly what the TAs should expect to see)

Implement other features

How does this project relate to material covered in CSC258?

FPGA board

Verilog code

VGA adapter

FSM

Counter

Clock?

Mux?

What's cool about this project (to CSC258 students and non-CSC258 students)?

Basic reinforcement learning neural network in FPGA board

Why does the idea of working on this appeal to you personally?

Build an AI with FPGA