ELECTRONIC ASSIGNMENT COVERSHEET

Murdoch UNIVERSITY

Student Number	33540698
Surname	Neve
Given name	Peter
Email	peterneve130@gmail.com

Unit Code	ICT206
Unit name	ICT206 Intelligent Systems
Enrolment mode	Internal
Date	October 15, 2021
Assignment number	2
Assignment name	Project
Tutor	Graham Mann

Student's Declaration:

- Except where indicated, the work I am submitting in this assignment is my own work and has not been submitted for assessment in another unit.
- This submission complies with Murdoch University's academic integrity commitments. I am aware that information about plagiarism and associated penalties can be found at http://www.murdoch.edu.au/teach/plagiarism/. If I have any doubts or queries about this, I am further aware that I can contact my Unit Coordinator prior to submitting the assignment.
- I acknowledge that the assessor of this assignment may, for the purpose of assessing this assignment:
  - reproduce this assignment and provide a copy to another academic staff member; and/or
  - submit a copy of this assignment to a plagiarism-checking service. This web-based service may retain a copy of this work for the sole purpose of subsequent plagiarism checking, but has a legal agreement with the University that it will not share or reproduce it in any form.
- I have retained a copy of this assignment.

I am aware that I am making this declaration by submitting this document electronically and by using my Murdoch ID and password it is deemed equivalent to executing this declaration with my written signature.

Signed Peter Neve

Reactive State Machine Boxer

Table of Contents

Acknowledgements

I thank Jayanam for developing the fundamental retreat code.
I thank Unity for providing the locomotion system code.
I thank all participants for taking part in the experiment.

Introduction

This project focused on creating a reactive state machine for a boxing robot using Unity. This entailed a finite state machine that is controlled by a reactive sensor for the opponent's glove positioning, a navigation agent for movement states, and an environment sensor to keep the robots away from the edge of the boxing ring. The project's application is a spectator game where users can modify parameters (health, stamina, resilience, etc.) and witness how it affects the match between the two robots.

The main goal of the project is to create an entertaining experience for users measured by a Game Experience Questionnaire. Secondary goals of the project are to demonstrate that the modifiable parameters affect the outcome of a match and explore how well the robots react to their opponent's actions (i.e. they hit or block in the correct direction).

Background

Finite State Machines (FSM) are commonly used in video games for developing non-player character (NPC) behaviour [1]. Past research has made use of a reactive system to control an FSM to create a more realistic NPC with promising results [2]. Navigation agents are also commonly used in NPC development to give them pathfinding abilities [3].

Boxing AI systems have been briefly explored in several different ways: one way only used Unity's machine learning agents to control a fighter [4]. The results were two fighters standing upright and waving their gloves frantically near their face with negligible recognition of what punches were thrown. Another paper used a neural network with reinforcement learning that taught itself boxing [5]. This approach resulted in recognizable boxing techniques such as jabs, dodges, hooks, etc. the only issue with the project in relation to the one in this paper is its advanced implementation. There are other implementations of boxing AI but only speculation of how it is made is possible due to a company owning the rights to it [6][7].

With the limited background information on actual boxing AI systems and the common use of Finite State Machines (FSM) in non-player characters, this project explored the implantation of an FSM with reactive agents, stamina, and health for controlling the state machine. The FSM utilizes realistic animations that are fixed in comparison to [5] which features more dynamic animations. The fixed animations still prove more effective than those present in [4].

AI Method and Tools

The project used Unity and Visual Studio as the final application designed to play as a game. Unity was not the only option as Unreal Engine was also a viable tool to complete the project; however, it would require significant learning to accomplish such a task in the available time. Unity was chosen as the primary tool for the project due to existing knowledge of all required systems and it has a built-in navigation system.

The AI methods used in the project are reactive agents, navigation agent, Finite State Machines (FSM), and a locomotion system. There were two reactive agents used to control the FSM, one's task is to track the opponent's gloves and record which direction they were coming and notify the animation state machine of which direction it should react in (left, right, or centre). The second reactive agent grants the AI knowledge of its surroundings and is responsible for keeping the fighter away from the ropes of the boxing ring. This was done by finding the closest edge of the navigation agent and checking if the distance was within the fighter stopping distance, thus triggering an escape state.

Unity supplied the navigation agent for both fighters enabling them to move towards/away from each other along with providing a collision system to prevent them from clipping within each other. The state machine controls the navigation agent via setting the target/evade destination in a chase/retreat state respectively. The navigation agent also controls the Finite State Machine through its stopping distance where if the opponent is within this distance the chase state is no longer needed, as they have reached the opponent.

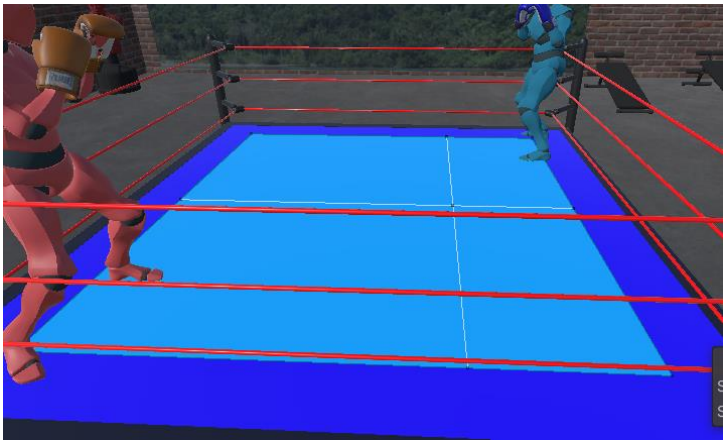


Figure 1: Shows the area in which the navigation agent can access in light blue

The solution features two Finite State Machines (FSM), one for animation and another for all functionality. Despite the separation, they are tightly coupled as the functionality FSM is used to control the animations FSM with some overlap, such as states: Chase, Retreat, and Escape all enable the Move state of the animation FSM. All animation states except victory reference a blend tree, figure 2 below, which groups animations together and plays them based on a set or single variable, table 1 below, where this variable is assigned by the locomotion system or functionality FSM.

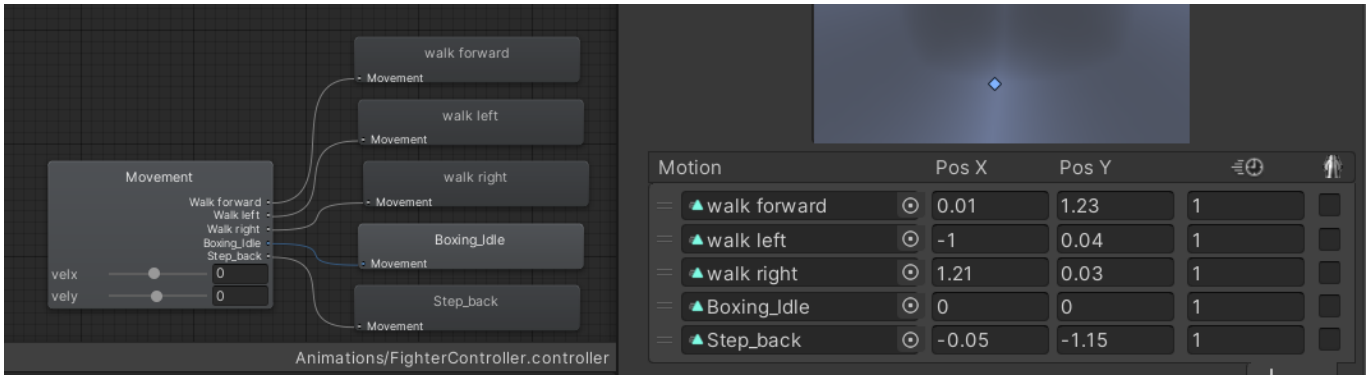


Figure 2: Blend tree for controlling movement animations based on velx and vely float variables

The biggest difference between the two FSMs is their transitions, see table 1 and figure 4, where the animation FSM is controlled by Boolean values and the blend trees, figure 2, are controlled by floating values.

Table 1: List of state transition parameters for animation state machine

Transition parameter	**Used by**
velx	Float that affects the movement animation tree
vely	Float that affects the movement animation tree
move	Boolean that allows for movement animation to play
attack high	Boolean that allows for high punch animation to play
Block	Boolean that allows block animations to play
Attack Type	Float that specifies which high/low punch animation will play
Block Type	Float that specifies which high punch animation will play
Attack low	Boolean that allows low punch animations to play
Defeat	Boolean that allows defeat animations to play
Defeat type	Float that specifies which defeat animation will play
Victory	Boolean that allows victory animation to play
Hit Type	Float that specifies which hit animation will play
Hit	Boolean that allows hit animations to play

The locomotion system is from [6] and is used to control the Move state of the animation FSM, figure 3 below, in such a way that it links the animation to the movement speed of the navigation agent. The locomotion system also came with a look at function to control the rotation of a character's head moving more realistically, although it is not highly noticeable in the project.

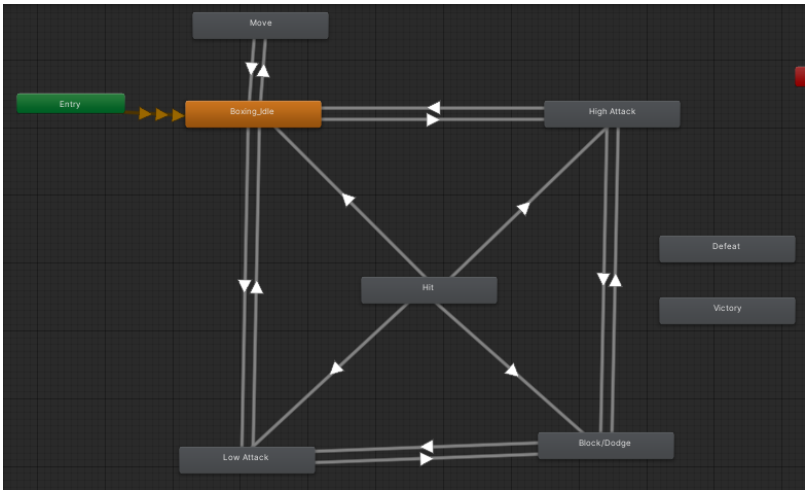
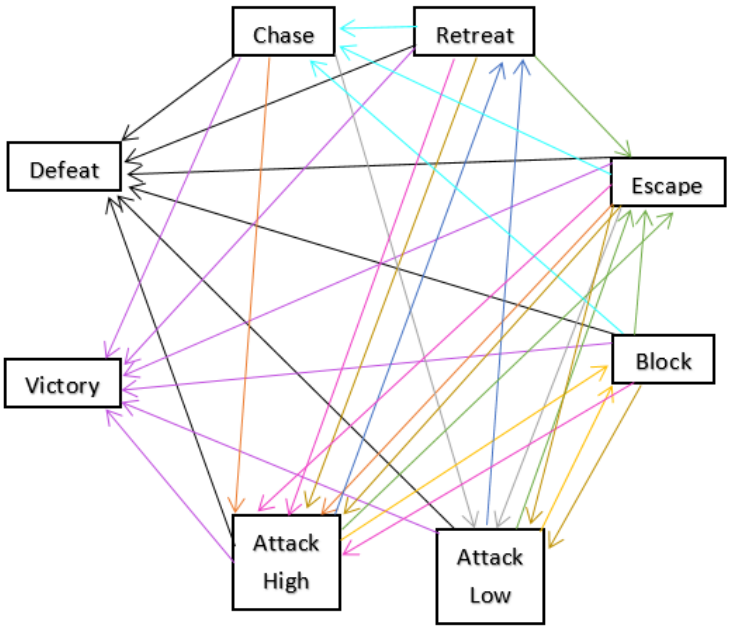


Figure 3: Simplified state transition diagram that both robots use to control animations

The functionality FSM, figure 4 below, is primarily controlled by a fighters' remaining health and stamina but is influenced by reactive sensors and the navigation agent. The escape state is only controlled by the reactive agent that sensors the environment, meaning without this sensor a fighter will not evade the ropes resulting in a corner fight.



Legend	
→	Opponent health <= 0
→	Health <= 0
→	N = 1
→	N = 0
→	R < block chance
→	R >= block chance
→	ropeDist <= stop dist
→	opponentDist > stop dist
→	resilience check stamina >= rand(0, fitness level) + highcost
→	resilience check stamina >= rand(0, fitness level) + lowcost

Resilience check
(Stamina >= stamina * rand(0, resilience) or
Health >= health * rand(0, resilience)

Figure 4: State transition diagram for functionality (brain) state machine

Evaluation & Conclusion

To evaluate the project against its goals 10 anonymous volunteers have tested the application and filled out a Game Experience Questionnaire that is broken into four key sections: in-game, post-game, parameter behaviour, environment sensor, and additional comments. The in-game section assesses the experience as scores in seven components: Immersion, Flow, Competence, Positive and Negative Affect, and Tension. Post-game assesses return to reality, tiredness, positive and negative experiences [5][6]. Parameter behaviour is designed to evaluate the impact of parameter modification on a game along with ranking the parameters on how much they affect the game. Environment sensor records how well the system worked in regards to effectiveness. Lastly, additional comments will allow participants to convey their thoughts about the project through a brief description with emotional wording.

Participants were given a copy of the user guide to learn how to use the game. Then advised to look at the questionnaire to understand what to look for while playing the game for as long as they needed to answer the questions accurately. The questionnaire was developed as a Google Form for automatic general data analysis of each question.

The questionnaire focuses on how the game made a participant feel and their opinion on how effective variables are in changing the game and how well the environment sensor keeps the fighter away from the ropes. All questions are ranked from 1 - 5 where one is not at all and five is extremely. The only exception is a question under the parameter behaviour section where participants must rank the adjustable variables in order of effectiveness (1 - 10) where one is most and ten is least.

The biggest challenge with the project was detecting the direction of the opponent's boxing gloves. This was made clear when trying to implement a dodge state into the functionality state machine; this state was left out due to its errors and lack of appropriate animations. Another challenge was the environment sensor and having it set a single target destination for the fighter that was away from the edge of the boxing ring. The sensor still does not function correctly as the fighter stays near the edge while being in the escape state however, they do move more resulting in a more entertaining fight.

A limitation of the evaluation of the project is the sample size and their demographic. For a more in-depth evaluation, there should be more participants with a varying backgrounds such as gamers, developers, etc. For future work on the project, it can adopt a reflective agent to find the optimal set of parameters without human interaction, along with refining the animations to increase realism.

Results

Figure 5, shows the average score for each categorized section of the in-game section of the questionnaire. This supports the main goal of entertainment by showing the game had a 70% positive effect on players with only around 42% feeling tension that can be explained by them favouring one robot over another, thus further supporting the entertainment goal.

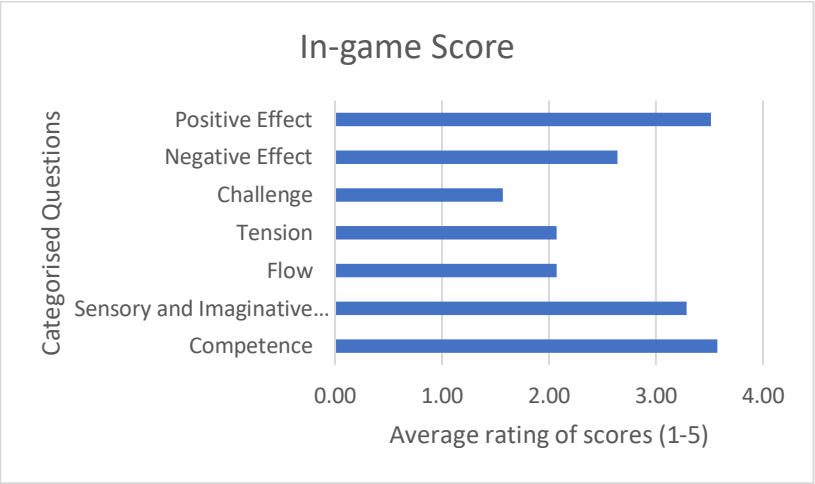


Figure 5: Bar graph combining questions into meaningful categories for in-game feelings

Figure 6, highlights the average score of questions according to measurable categories (return to reality, tiredness, positive experience, negative experience). The graph indicates participants are left in a more positive mood with reduced tiredness. However, it also shows that they did not find it difficult to return to reality. This supports the main goal of entertainment by leaving players in a better mood when finished.

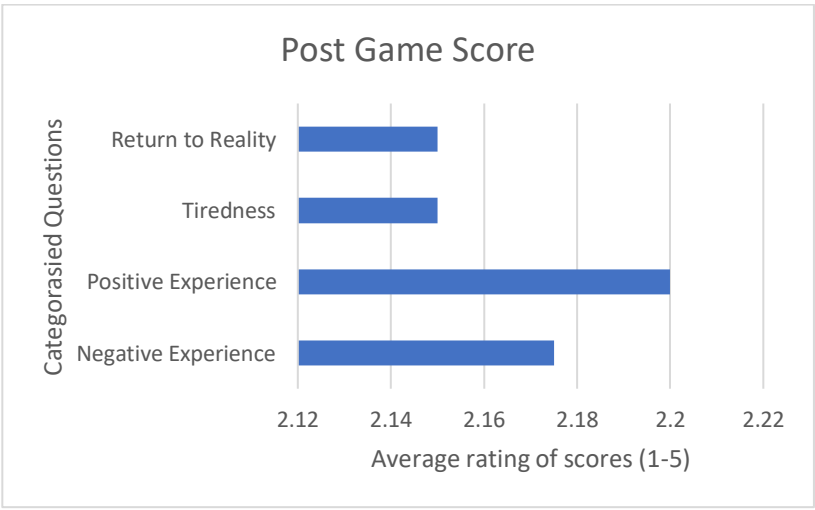


Figure 6: Bar graph combining questions into meaningful categories for post-game feelings

Figure 7, focuses on the effectiveness of the environment reactive agent and explores how effective it was at keeping the fighter away from the ropes. Around 70% of participants felt that it performed well. Thus supporting the secondary goal in terms of the robots being affected by the reactive agents.

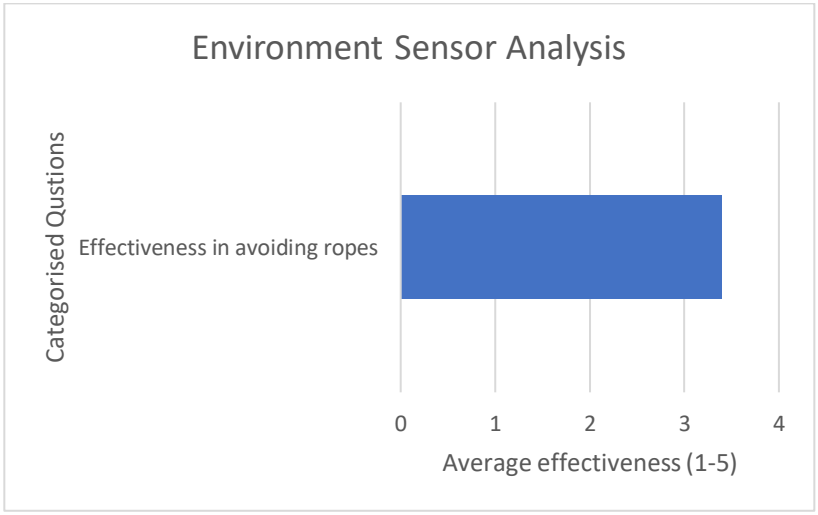


Figure 7: Bar graph depicting the average effectiveness of the environment sensor according to surveyed questions

Figure 8, supports the secondary goal of how changing variables affect the outcome of a match and whether some were stronger than other variables. 96% of participants felt that the variables did affect the outcome of a match while 8% thought the variables were balanced in their effect. Thus proving the supported secondary goal to be true.

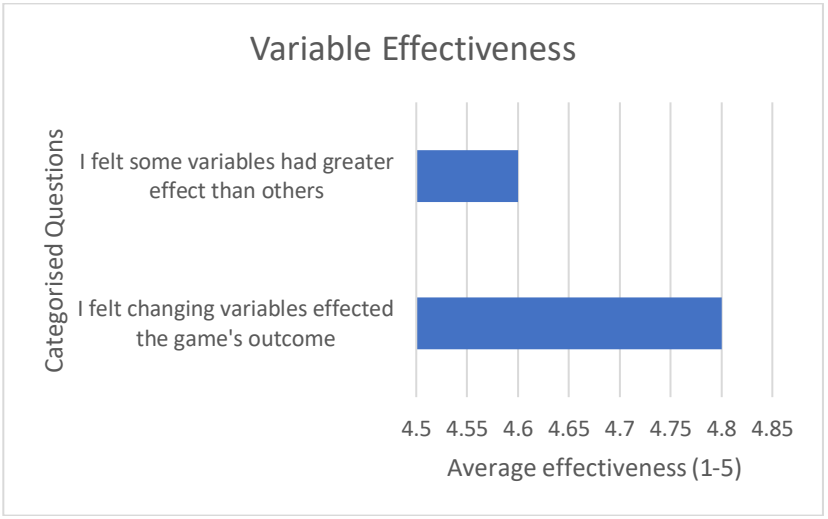


Figure 8: Bar graph depicting the average effectiveness of variables according to surveyed questions

Table 2, shows what parameters had the greatest impact on a match. Participants believed strength to be the most effective and the environment sensor to have the least effect on a match. There are also several parameters where participants felt they had the same impact such as health and stamina, recovery rate, and health recovery.

Table 2: Most common ranks of variable effectiveness on a match

Parameter Name	Most Common Rankings
Health	2
Stamina	2
Strength	1
Block Chance	7
Recovery Rate	4
Stamina Recovery	8
Health Recovery	4
Fitness Level	9
Resilience	8
Environment Sensor	10

Conclusion

In conclusion, the project was a success as 70% of the participants found it entertaining and both secondary goals gained supporting evidence. This result was above expectations as the final application remains to be desired. The project explored how finite state machines are used and how many things such as changing variables, reactive sensors, and navigation agents can control them. Overall, the project can be improved through further constraints on parameters and behaviours such as reducing stamina regeneration as the match continues. In the end, the project was a success as it produced an entertaining application and showed one way of how Finite state machines can be used and controlled.

References

[1] H. Lou and S. McArdel, *AI in Video Games: Toward a More Intelligent Game - Science in the News*, Science in the News, 2021. [Online]. Available: https://sitn.hms.harvard.edu/flash/2017/ai-video-games-toward-intelligent-game/. [Accessed Oct. 14, 2021].

[2] L. Gruenworldt, M. Katchabaw and S. Danton, *CREATING REACTIVE NON PLAYER CHARACTER ARTIFICIAL INTELLIGENCE IN MODERN VIDEO GAMES*, Proceedings of the 2005 GameOn North America Conference, 2005. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.505.749&rep=rep1&type=pdf. [Accessed Oct. 14, 2021].

[3] P. Harsadi, S. Asmiatun, A. N. Putri, *Dynamic Pathfinding for Non-Player Character Follower on Game*, Jurnal Teknik Informatika CIT Medicon, 2021. [Online]. Available: https://doi.org/10.35335/cit.Vol13.2021.68.pp51-58. [Accessed Oct. 14, 2021].

[4] mbaske, *ml-selfplay-fighter: Self-Play Boxing Match made with Unity Machine Learning Agents*, GitHub, 2021. [Online]. Available: https://github.com/mbaske/ml-selfplay-fighter. [Accessed 10- Oct- 2021].

[5] J. won, D. Gopinath and J. Hodgins, *Control Strategies for Physically Simulated Characters Performing Two-player Competitive Sports*, Facebook Research, 2021. [Online]. Available: https://research.fb.com/publications/control-strategies-for-physically-simulated-characters-performing-two-player-competitive-sports/. [Accessed Oct. 14, 2021].

[6] *Thrill of the fight*, Sealost Interactive LLC, 2016. [Online]. Available: https://store.steampowered.com/app/494150/The_Thrill_of_the_Fight__VR_Boxing/. [Accessed Oct. 10, 2021].

[7] Servios, *Creed: Rise to Glory*, Surcois and Sony Interactive Entertainment, 2018. [Online]. Available: https://store.steampowered.com/app/804490/Creed_Rise_to_Glory/. [Accessed Oct. 10, 2021].

[8] Unity, *Coupling Animation and Navigation*, Unity Documentation, 2020. [Online]. Available: https://docs.unity3d.com/Manual/nav-CouplingAnimationAndNavigation.html. [Accessed Oct 10, 2021].

[9] W. A. IJsselsteijn, Y. A. W. de Kort and K. Poels, *The Game Experience Questionnaire*. Eindhoven University Technology. 2013. [Online]. Available: https://eclass.uoa.gr. [Accessed Oct. 14, 2021].

[10] J. Högberg, J. Hamari and E. Wästlund, *Gameful Experience Questionnaire (GAMEFULQUEST): an instrument for measuring the perceived gamefulness of system use*. 2019. [Online]. Available at: https://doi.org/10.1007/s11257-019-09223-w. [Accessed Oct. 14, 2021].

[11] Jayanam, *Unity Navmesh AI Tutorial : Flee from Enemy*, YouTube. 2018. [Online]. Available: https://www.youtube.com/watch?v=ZjIg9F3FRJs. [Accessed Sept. 20, 2021].

Appendix - Related files

Table 3: List of related files externally attached to the report

Source Code
FighterAI
EnvironmentSensor
FighterHealth
Menu
ReactiveSensor
TimeControl
CameraClass
InvokeButton
Randomize
SliderValueToText
Questionnaire
GE_Questionarre
Raw data
RSMBoxer_GEQ (Responses)

User Guide

Reactive State Machine Boxer is a spectator game where users can watch two robots duke it out with customizable parameters that affect the match.

How to Run

To run Reactive State Machine, open the project folder and run the .exe file.

Game Interaction

Interaction in the game is limited as the player only controls the parameters of the AI's camera position and speed of time within the game. The primary function of the game is to act as a viewing medium for users to witness a reactive state machine AI agent fight another. Players can also pause a match to change settings, restart, quit or resume.

Movement

Players can stop the pan motion in the Main Menu, and when playing they can rotate left/right or re-enable pan mode. The camera always looks at the center of the boxing ring.

Key Bind	Purpose
P	Enable/disable auto rotation during the game or main menu
A / Left Arrow	Move camera to the left
D / Right Arrow	Move camera to the right

User Interface

Main Menu

Initial screen to the game.

- Play: start the match with defined settings
- Settings: Opens settings menu where you can modify a robot's parameters.
- Quit: Quits the application

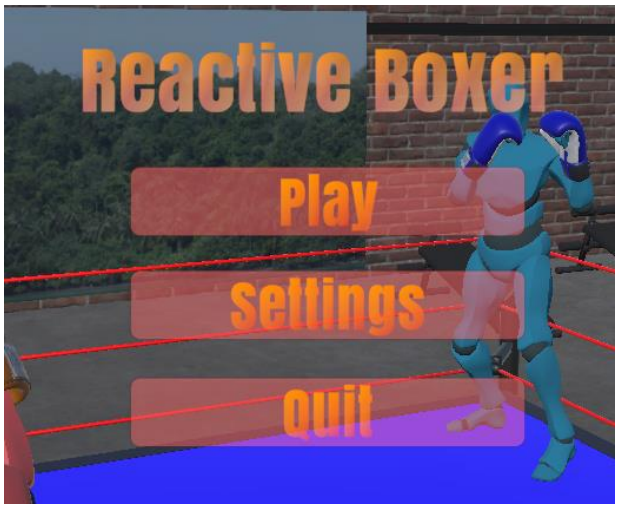


Figure 9: Screenshot of the main menu for reference

Settings
User interface menu that allows users to edit a robot's parameters, creating a different experience

Table 4: Parameter Settings Breakdown

Adjustable Parameter	Purpose
Health	Increase/Decrease robots max health
Stamina	Increase/Decrease robots max Stamina, Limits the number of consecutive punches
Strength	Increase/Decrease robots Strength
Block Chance	Increase/Decrease the chance for the robot to Block or Retreat to recover stamina/health (High = greater block chance, Low = greater chance to retreat)
Recovery Rate	Determines the recovery cycle length a robot has to wait until recovering health/stamina in seconds (High = slow recovery)
Stamina Recovery	Defines the amount of stamina a robot recovers every recovery cycle (defined by recovery rate)
Health Recovery	Defines the amount of health a robot recovers every recovery cycle (defined by recovery rate)
Fitness Level	max amount of remaining stamina a fighter can have before stopping their attack
Resilience	Defines the amount of consecutive health/stamina a robot can use at one time. The amount of health/stamina to recover is also based on this value.
Environment Sensor	Enables an escape state for the robot, this is triggered when they get close to the ropes.

- Reset: Resets individual variable values to default
- <-: copies blues' variables to red
- ->: copies reds' variables to blue
- Save: saves the customized values and return to the Main Menu
- Cancel: discards all changes, including prior saved changes
- Rand: randomizes all variables
- Reset all: Resets all variables to default values

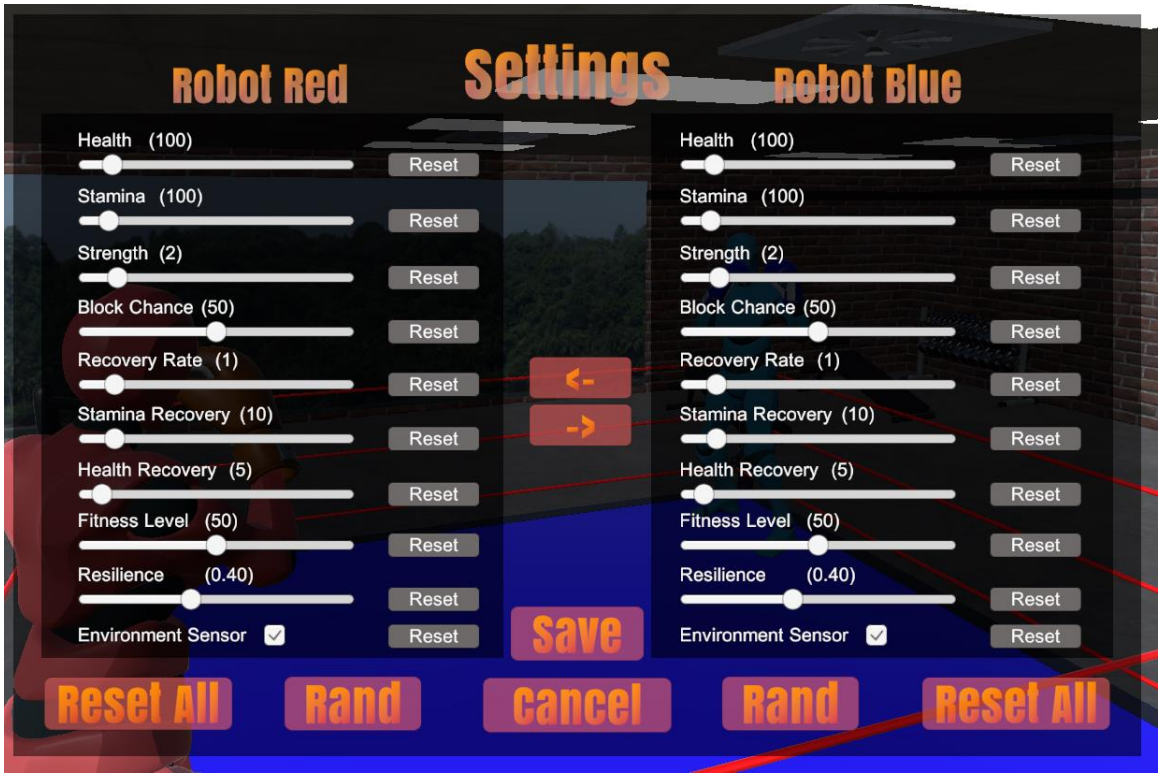


Figure 10: In-Game settings menu to act as a reference for table 1 (above)

Pause Menu
Grants users the ability to perform actions after they started a game.

- Resume: Continue the current match
- Restart: restarts the match with the same settings.
- Main Menu: Takes the user back to the main menu where they can edit settings.
- Quit: Closes the application

Key Bind	Purpose
Esc	Pause/Resume the game

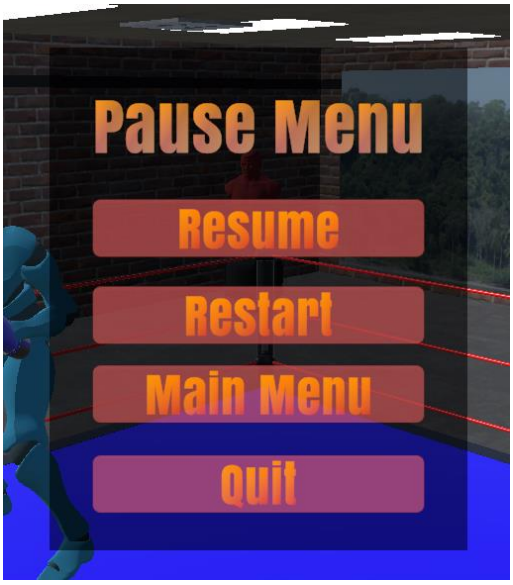


Figure 11: Graphic depiction of the pause menu

Heads up Display
Displays the robot's health, stamina, and current state along with a slider that controls the game's speed.

- Red bar: Health
- Orange bar: Stamina
- White field: Current state
- Timescale: Controls the speed of the game
- Time: Duration of match

Figure 12: In-game HUD showing both robots health, stamina and state along with game speed control