

Computer Games Development CW208

Technical Design Document

Year IV

Joshua Dunne

C00241588

|  |  |
| --- | --- |
|  |  |
| 12/12/2021 | |

# 

Contents

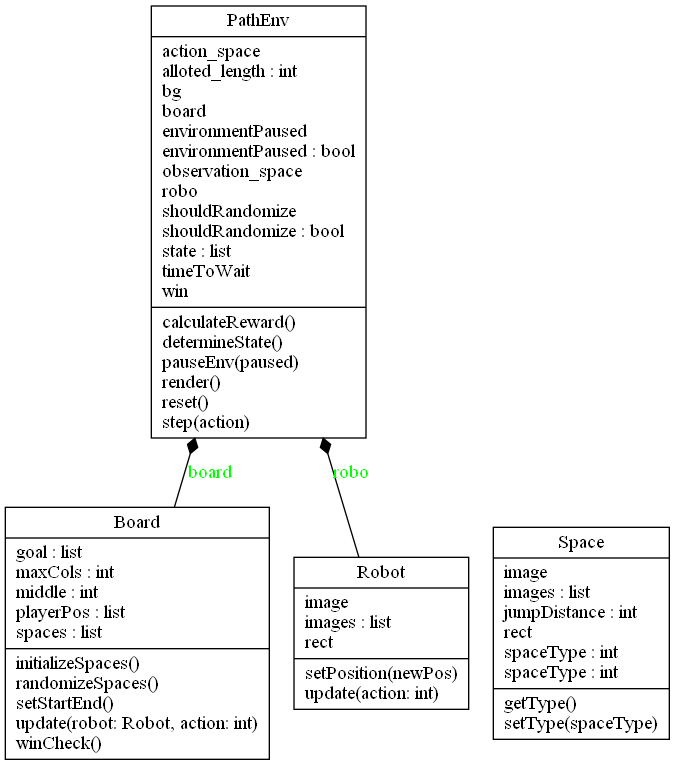
[**Technical Design**](#_heading=h.gjdgxs) **2**

[**References**](#_heading=h.u2v46im2tre5) **3**

# Technical Design

For my custom environment, the Board will be a 2d array of Spaces. This 2d array will be stored within the board class, alongside the Player’s position on the Board, and the Goal.  
The Board will create the 2D array, filling it with Spaces until a specified max column length that is passed into the constructor.

Each Space will be a simple class that holds an image dictating what type it is visually. The type they are is dictated by what amount they push the Player on the board and can be changed manually to dictate the start/goal spaces.

The Player’s Robot will be represented independently from the board but will still adhere to the Board’s position. The Robot will tell the Board which way it wants to move and will update its visual to face the correct direction. The Board will figure out where the Robot should be placed and place the Robot in that space.  
If the Robot is over the Goal, the environment ends.

Text

Description automatically generated with low confidence

Text

Description automatically generated with low confidence

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

The Action Space of the environment is a Discrete with a range of 4. This means it can generate a number from 0 -> 3 inclusive. Each of these numbers dictate what action has been chosen, with 0 being Right, 1 being Left, 2 being Up and 3 being Down.  
  
The Agent uses an Action picked from this Action Space during it’s prediction to interact with the environment, which generates an Observation Space after the Action has been applied.

The Observation Space consists of 5 numbers. The first number is a Discrete of 80, and the remaining numbers are Discretes of 5.  
  
The first number tells the Agent what space they are on. It is calculated by adding the X position of the Robot to the Y Position of the Robot times the Max number of Columns on the Board. This means that if the Robot is at 0,0, the number will be 0. If the Robot is at 1,1, the number will be 1 + Max Columns, and so on.  
  
The remaining 4 numbers tell the Agent what space type is directly up, down, left, and right of its current position. This was implemented to give the Agent a sense of vision, as it can see what it is exactly around it.   
This helped the Agent greatly when it came to later training, as it could use the information for what spaces are around it to determine what to do, rather than an earlier implementation I had made where it only told the Agent where it was on the board and nothing else.

# References

*[1] Renotte, Nicholas. “Reinforcement Learning in 3 Hours | Full Course Using Python.” YouTube, 6 June 2021,* [*https://www.youtube.com/watch?v=Mut\_u40Sqz4*](https://www.youtube.com/watch?v=Mut_u40Sqz4)*.*

*[2] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, & Noah Dormann (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research, 22(268), 1-8.*

*[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, & Wojciech Zaremba. (2016). OpenAI Gym.*

*[4] Tewari, U. (2020, April 14). Which reinforcement learning-RL algorithm to use where, when and in what scenario? Medium.Datadriveninvestor.Com. Retrieved January 27, 2022, from* [*https://medium.datadriveninvestor.com/which-reinforcement-learning-rl-algorithm-to-use-where-when-and-in-what-scenario-e3e7617fb0b1*](https://medium.datadriveninvestor.com/which-reinforcement-learning-rl-algorithm-to-use-where-when-and-in-what-scenario-e3e7617fb0b1)

*[5] Base RL Class — Stable Baselines3 1.5.1a4 documentation. (2020, June 9). Stable-Baselines3.Readthedocs.Io. Retrieved January 27, 2022, from* [*https://stable-baselines3.readthedocs.io/en/master/modules/base.html*](https://stable-baselines3.readthedocs.io/en/master/modules/base.html)

*[6] Moni, R. (2021, December 7). Reinforcement Learning algorithms — an intuitive overview. Medium. Retrieved January 27, 2022, from* [*https://smartlabai.medium.com/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc*](https://smartlabai.medium.com/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc)

*[7] Simonini, T. (2019, February 5). Proximal Policy Optimization (PPO) with Sonic the Hedgehog 2 and 3. Medium. Retrieved January 27, 2022, from* [*https://towardsdatascience.com/proximal-policy-optimization-ppo-with-sonic-the-hedgehog-2-and-3-c9c21dbed5e*](https://towardsdatascience.com/proximal-policy-optimization-ppo-with-sonic-the-hedgehog-2-and-3-c9c21dbed5e)

*[8] freeCodeCamp.org. (2018, March 29). An introduction to Deep Q-Learning: let’s play Doom. Retrieved January 27, 2022, from* [*https://www.freecodecamp.org/news/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8/*](https://www.freecodecamp.org/news/an-introduction-to-deep-q-learning-lets-play-doom-54d02d8017d8/)

*[9] Reinforcement Learning Tips and Tricks — Stable Baselines 2.10.2 documentation. (n.d.). Stable-Baselines.Readthedocs.Io. Retrieved January 27, 2022, from* [*https://stable-baselines.readthedocs.io/en/master/guide/rl\_tips.html#which-algorithm-should-i-use*](https://stable-baselines.readthedocs.io/en/master/guide/rl_tips.html#which-algorithm-should-i-use)

*[10] Wikipedia contributors. (2022, April 12). Multilayer perceptron. Wikipedia. Retrieved March 24, 2022, from* [*https://en.wikipedia.org/wiki/Multilayer\_perceptron*](https://en.wikipedia.org/wiki/Multilayer_perceptron)

*[11] An Overview on Multilayer Perceptron (MLP). (2022, February 21). Simplilearn.Com. Retrieved March 24, 2022, from* [*https://www.simplilearn.com/tutorials/deep-learning-tutorial/multilayer-perceptron*](https://www.simplilearn.com/tutorials/deep-learning-tutorial/multilayer-perceptron)

*[12] Miyoung Han. Reinforcement Learning Approaches in Dynamic Environments. Databases [cs.DB]. Télécom ParisTech, 2018. English. tel-01891805*

*[13] R. S. Sutton and A. G. Barto. Introduction to Reinforcement Learning. MIT Press, 1998.*

*[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.*

*[15] PPO — Stable Baselines3 1.5.1a4 documentation. (n.d.). Stable Baslines 3. Retrieved January 27, 2022, from https://stable-baselines3.readthedocs.io/en/master/modules/ppo.html*