# Term 1 Report:

## Section A:

Evolving Virtual Creatures is about experimentation with computational intelligence in order to design a device (creature) that may iteratively improve efficiency towards completing a task. Is the human body the most efficient implementation of a body that can carry out tasks that are becoming invariably more complex or difficult in the human world? Are current technologies used on production lines or industry, the best use of electricity or can complete tasks with the lowest cost? What about AI in interactive software or games? Are they the most realistic and competent that they can be?

This project is designed to make a start on producing what may later be used in searching for the most efficient actions or creature for particular tasks. Later objectives may involve subjecting the evolutionary algorithms explored in this project to particular tasks that may have real world application but for this project, the results will not be so complex due to the lack of physical, logistical and schematical (i.e. understanding how a creature should move but not mechanically how without an engineering insight.) data due to the nature of the graphical simulations that harness pre-existing, non-specialised 2D and 3D physics engines.

This project's aim is to exhibit evolutionary behaviour and development of a virtual creature within a virtual space. This includes creating a virtual testing space in order to trial the creatures; the creatures that will inhabit this virtual test space with the ability to initially move along the x and y plane, with later developments including the z plane; A variety of simple tasks the creatures will be subject to in order to observe their efficiency and the improvements where they exist. This will all rely on producing an evolutionary algorithm to house the virtual development of a virtual organism that is based on the principle of random search – with effective implementation, this will ensure that no single strain of genetics will be repeatedly generated per case.

This will be demonstrated by the virtual creature's performance on 3 predetermined tasks; Travelling from point A to B successfully and in a timely manner; tracking efficiency of the creature as the time frame shortens. Once complete, obstacles will be introduced to make travelling from A to B more difficult but will ultimately display iterative improvement in path finding and creature evolution. Once these two tasks are demonstrated, the final task will be for the creature to collect an object from point B and travel to point C. This in concept will hybridize the creature in order to create an efficient multipurpose creature without compromising on speed or efficiency i.e. path finding and object transportation in place of one of the two in this concept.

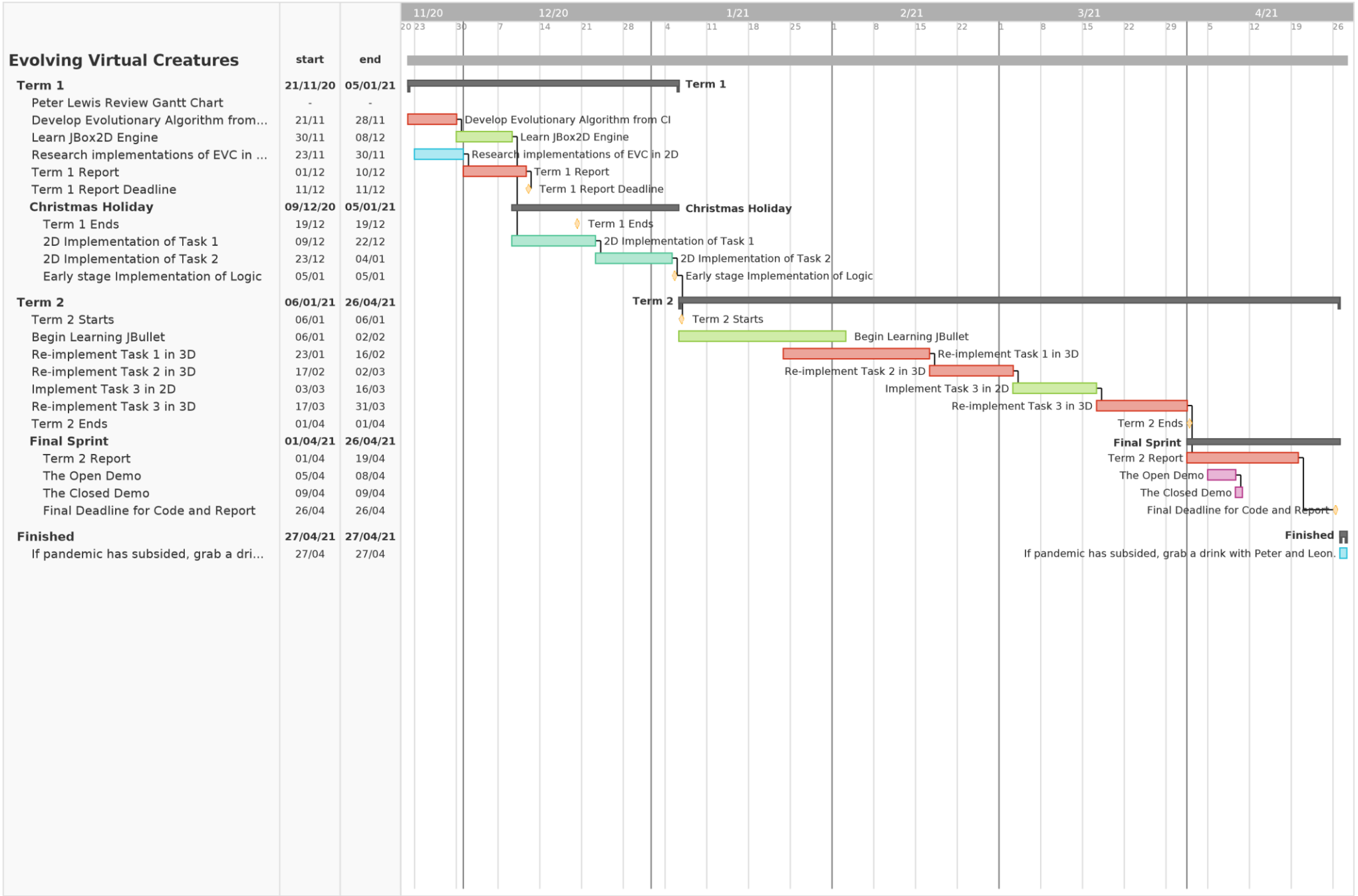
With later developments of this project including extra layers of complexities; The end result should ultimately allow us to develop highly capable creature designs for a desired set of

tasks that it will be subject to; with many autonomous equipment being locked down to having a singular use, this testing space should allow us to re-evaluate machine autonomy from production lines to perhaps power generators that supply electricity to our cities. This project is inspired by Karl Sims' Evolution Simulation where the virtual space allowed virtual organisms to evolve, reproduce and the results of these evolutions were pitted against each other in order to complete a task much like how this project plans to. However, in Sims' project, Sims was not particular about the creatures that were made to compete to complete a simple task.

In this project, the aim is to be able to produce competencies in the creature under evolution. With these competencies, it will be possible to produce a creature that is specifically designed for that task. Later iterations will allows industry or digital businesses to test their own designs for an efficient creature in this evolutionary work space and observe what improvements could be made to them.

The research and learning involved in deciding on this project, which direction to take it and how, are based on the following:

- CS3910 Computational Intelligence – Basis for intellectual recombination and evolutionary logic, when understood become manipulable for this project.
- Karl Sims' Evolved Virtual Creates (https://www.karlsims.com/evolved-virtual-creatures.html) – Providing the base inspiration for this project.
- Physics Engines (Box2D, JBox2D, JBullet for 3D) – A bespoke physics engine that more closely resembles real world physics may have helped to produce more accurate results to take away for real world application. However, being short on time forces the project to use pre-existing physics engines that replicate physics for the context of animation or games development leaving real world data to be less important.
- JBox2D - https://github.com/jbox2d/jbox2d
- JBullet - http://jbullet.advel.cz/
- Introduction to Evolutionary Programming (A.E. Eiben J.E. Smith. Introduction to Evolutionary Programming) – General knowledge of evolutionary techniques.
- A Flexible and Simplified 2D Environment for Evolving Autonomous Virtual Creatures (Ricardo Sisnett Riot Games Inc, Los Angeles, U.S.A. https://www.scitepress.org/Papers/2015/56179/56179.pdf)
- BoxCar2D (http://boxcar2d.com/about.html) – An interactive evolutionary application which allows the user to test different models for a vehicle to see how well it does and if it can evolve to produce better results.
- TeamGantt – Task scheduling application (https://www.teamgantt.com/).




| Evolving Virtual Creatures | start | end |
|---|---|---|
| **Term 1** | **21/11/20** | **05/01/21** |
| Peter Lewis Review Gantt Chart | - | - |
| Develop Evolutionary Algorithm from... | 21/11 | 28/11 |
| Learn JBox2D Engine | 30/11 | 08/12 |
| Research implementations of EVC in ... | 23/11 | 30/11 |
| Term 1 Report | 01/12 | 10/12 |
| Term 1 Report Deadline | 11/12 | 11/12 |
| **Christmas Holiday** | **09/12/20** | **05/01/21** |
| Term 1 Ends | 19/12 | 19/12 |
| 2D Implementation of Task 1 | 09/12 | 22/12 |
| 2D Implementation of Task 2 | 23/12 | 04/01 |
| Early stage Implementation of Logic | 05/01 | 05/01 |
| **Term 2** | **06/01/21** | **26/04/21** |
| Term 2 Starts | 06/01 | 06/01 |
| Begin Learning JBullet | 06/01 | 02/02 |
| Re-implement Task 1 in 3D | 23/01 | 16/02 |
| Re-implement Task 2 in 3D | 17/02 | 02/03 |
| Implement Task 3 in 2D | 03/03 | 16/03 |
| Re-implement Task 3 in 3D | 17/03 | 31/03 |
| Term 2 Ends | 01/04 | 01/04 |
| **Final Sprint** | **01/04/21** | **26/04/21** |
| Term 2 Report | 01/04 | 19/04 |
| The Open Demo | 05/04 | 08/04 |
| The Closed Demo | 09/04 | 09/04 |
| Final Deadline for Code and Report | 26/04 | 26/04 |
| **Finished** | **27/04/21** | **27/04/21** |
| If pandemic has subsided, grab a dri... | 27/04 | 27/04 |

## Section B:

The first term of the academic year has been difficult but there has been a lot that I have learnt. Over the weeks, I established a connection with my Final Year Project (FYP) Supervisor (My Soup) and Personal Tutor, Dr Peter Lewis, frequently keeping him up to date on my progress and findings. The first thing Peter and I discussed was, what did I want to take home this year from my final year project? I had an initial concern of dread, feeling that the project I chose; while I thought it was incredibly interesting, was far out of my abilities. After much discussion, I left with a sense of confidence that it was better to do something incredibly interesting, sufficiently rather than something boring, very well.

Peter, and I mapped out a range of ideas before deciding on the current design prospective, as my initial ideas were seemingly elaborate and ostentatious; desiring for my Evolving Virtual Creatures (EVC) to carry out complex tasks before learning to crawl. The current design prospective is as described above. We then began discussing how to implement this project and what tools may be useful to utilize. As my project involved creating 3D EVC and my preference for programming language was Java; Peter, recommended looking into JBullet, for an easily integrable 3D Java physics engine. Peter, also suggest that it would be simpler to implement the logic required for EVC in a 2D realm before transitioning over to 3D. The rational perspective for this was that, provided the language and variables largely remained the same, the only difference would be the 3D engine that I would need to plug into the program with a few extra tests to accommodate the extra variables that would be created as a result of working with a 3$^{rd}$ plane.

One of my choices of modules for the final year was Computational Intelligence (CI). Unknown to me that this would be the best choice of modules I would choose as the intelligent logic and algorithms of multiple approaches would be taught to me. Once I was able to understand how to completely produce these algorithms, I would be able to simply take an algorithm and make adjustments for my project. Much of the work done this term was to focus on logic, plan and document my progress and to eventually be at a stage where I was able to begin the implementation of the first task.

Soon after my initial meeting with my soup, I began looking into forums to see whether Box2D is well supported and that any issues flagged were easily resolvable. In the process of this, I learnt that Box2D was a C# engine and began to look for alternatives. My assessment of Box2D is; while Peter recommended it with a great deal of confidence, I could either learn to code in C# and then look for an alternative 3D physics engine, or I could look for a java port. Multiple developers since the release and popularisation of Box2D ported multiple versions of multiple languages on Box2D and I pulled a version from GitHub called JBox2D, developed in 2013 with ongoing support.

For around 2 weeks, I faced dependency issues when importing JBox2D into the Eclipse IDE. The package is a Maven project which involves creating dependencies with the Maven

compiler. Maven is a tool that is used to make creating dependencies in java code easier for you in the sense that, creating the class or object dependencies isn't required since Maven will do this for you. In order for Maven to do this, the program will have to install a helper tool in order to manage these dependencies automatically. The issue I faced was simply that the connector I was using to fetch the helper tool was either corrupt, or as many suggested on GitHub, the link to the Maven Catalogue was outdated and no longer supported. After multiple attempts to reinstall the entire project and update the catalogue link to no avail, I instead opted to uninstall eclipse and try again from scratch. I am unable to explain why, but reinstalling eclipse alone was the only troubleshooting that was required in the end. JBox2D was now fully integrated into my Java IDE.

I began testing the pre-existing test bed functions in order to see what functionality exist already in the engine and find places I could perhaps manipulate code for my own use. Unfortunately, due to other academic commitments, I placed further testing and learning of JBox2D on hold. This was near the reading week mark with the first of the CI deadlines fast approaching. I focused for the next 2 weeks on completing the sign off tasks for the coursework; understanding and implementing random search, 2-opt swap and swarm intelligence. Unfortunately I was unable to complete the evolutionary algorithm in time for the sign off but this was not a bad thing in my opinion as I wished not to rush this algorithm and know that the implementation was absolutely fleshed out and correct.

Post reading week was when I sat down to begin making notes for this report and produced a Gantt chart for the rest of my project scope. The reason for developing the Gantt chart now and not in the past was not solely just laziness. Up till now, I had an understanding of what I had to do in order to realise my FYP but not the confidence in my expertise in order to achieve them within a reasonable allotted time. Programming the local search algorithm may be a 1 day job for a confident and skilled programmer yet for me it took 3 days. How was I able to assess a reasonable amount of time to allot to a task? Much like what was written at the start of Section B, my soup helped to visualize a time frame each task may take given what the finer details of those tasks may involve and to assess from those whether the task itself is complicated or not. Another useful thing Peter mentioned was to work backwards in milestone of what needs to be done. With these things in mind, I completed the Gantt chart and began the rest of my tasks.

Around 2 weeks to the end of Term 1, I had completed the Evolutionary Algorithm, albeit a novel crossover-only implementation. Sufficient for the latter part of the CI coursework but not enough for the EVC project. Soon after this reports submission, I will have completed a more versatile version of the algorithm, featuring a hybridized mutation technique.