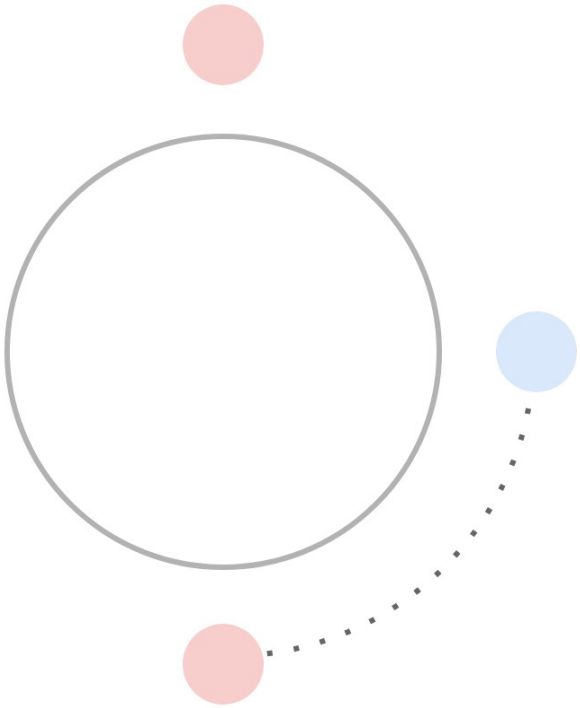


Writing, Designing and Implementing Electron

G54GAM - Coursework 2

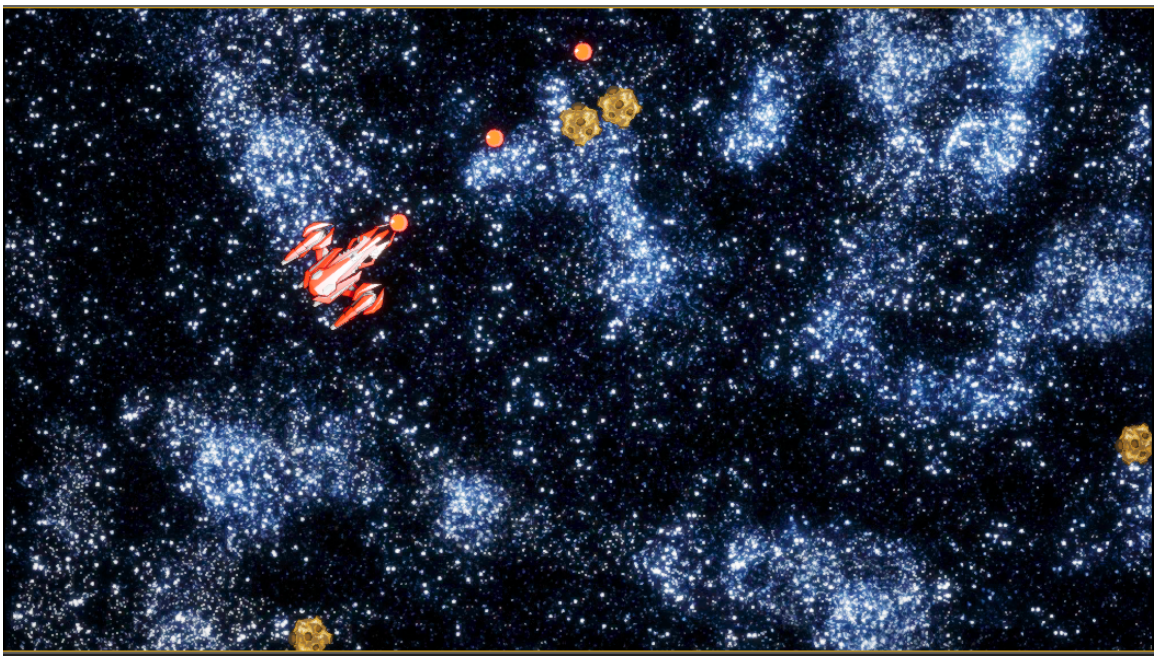
Dominic Jomaa
4256189

Dr. Martin Flintham

Lab Exercises Break Down

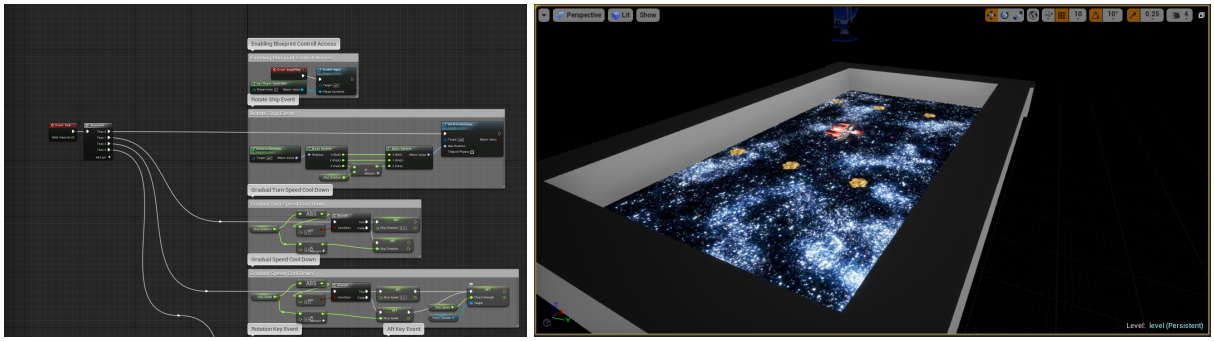
Exercise 1

This lab helped introduce many initial concepts associated with building a simple action game, such as how to construct a player controlled entity, how to handle collision events and then use those entities to build up some more game play-esque concepts such as beating a level of asteroids.



Lab 1 - Screenshot of Spaceship Asteroid Game

The level consisted of a simple sprite backboard of a space texture, and slightly above this a random assortment of asteroid elements each with their own initial launch vector. The player entity was a spaceship sprite with the ability to self rotate and the ability to move forward with the application of a short impulse in the direction the ship is pointing. The border of the level was built up of a simple arrangement of blocks that would stop any asteroid elements falling off the edge of the world and caused them to bounce back into the level space to be destroyed.



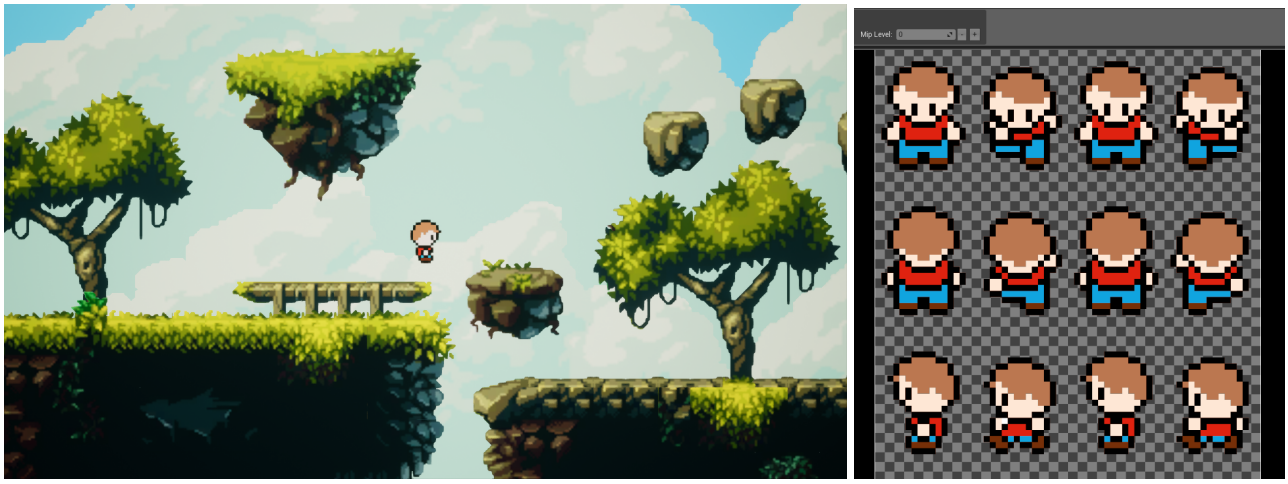
Lab 1 - Screenshots of Player Controller Blueprint and of Level Layout

Successes a When playing the game with others in the lab, criticisms were made of the various parameters of within the model of the game. Fortunately the parametric control over the game elements such as asteroid speed, player firerate and rotation speed all allowed the game to be focused and tailored to better suit the style and meet the suggestions of others.

Failures The unfamiliarity with the blueprint system meant much of desired changes never worked quite as expected such as to add an enemy npc to the game which would rotate to the player position but would do so either too quickly and overshoot or occasionally spin so fast that it would never slow down to face the player.

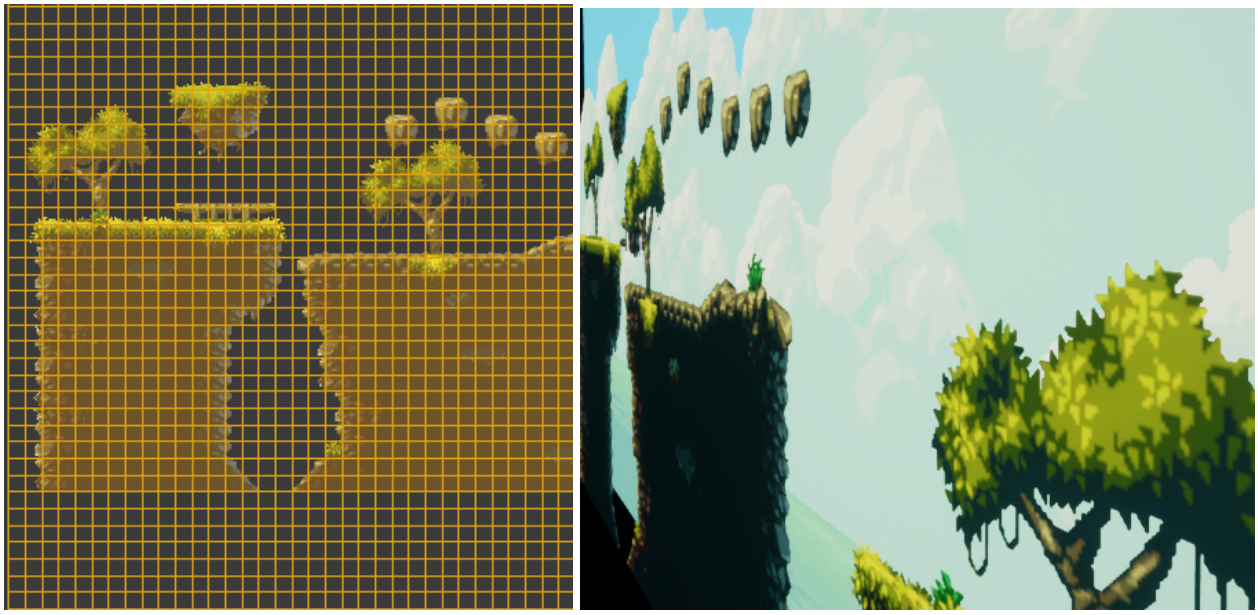
Exercise 2

This task was a first look into 2d platforming within unreal and taught the basics of using an appropriate camera to compress the space across the level, utilise a built in character animation system that allowed for the use of a sprite sheet and a further look into collisions which were necessary for walking and jumping on platforms.



Lab 2 - Player Character Jumping from Floor to Platform and Player Sprite Sheet

The level was built from a sprite sheet referenced as a lab asset, where each tile was hand detailed with a collision box and then arranged in the level using a tile map builder. The character assets was similarly imported and used to build a set of unreal flipbooks to visualise moving left, right, jumping and idling. The floating elements were positioned in the foreground along with the ground elements, some were positioned in the background and some in front of the foreground to give a sense of depth when moving around the level.



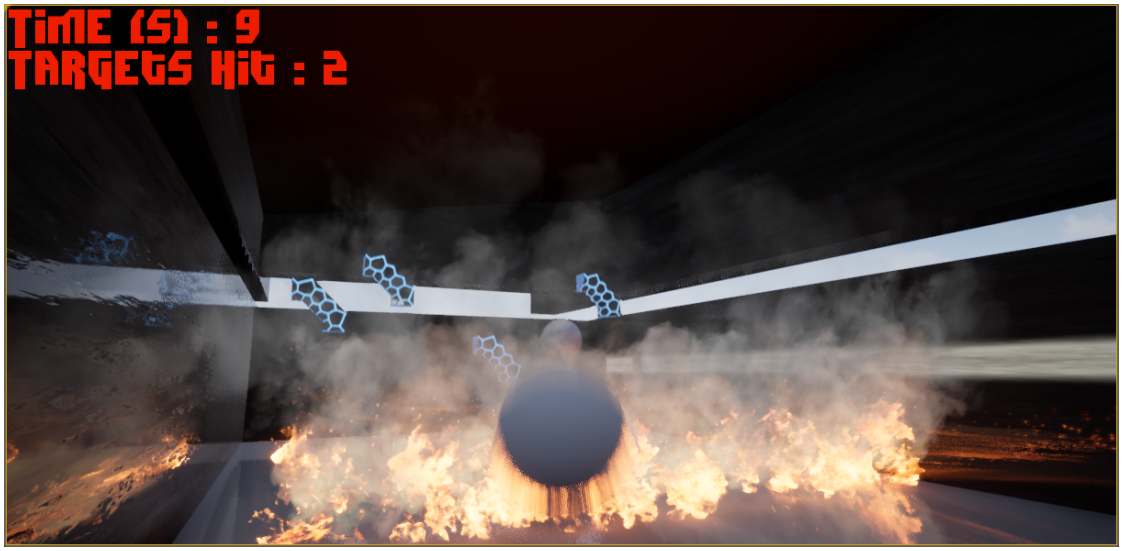
Lab 2 - Screenshot of Level Tile Map Builder and Uncompressed Platorming Space

Successes The level builder allowed for a large degree of creative freedom when laying out a how level and should look and feel while playing. The floating platforms were added to provide extra platforming freedom and decision on how to navigate to the end of the level, they also included a blueprint functionality which would allow them to oscillate up and down to add extra challenge.

Failures A greater focus could have been placed on building a level that worked with the tileset and with the platforming style in general, as the level structure never strayed very far from the root of the exercise. If approached again a level should be designed before attempting any implementation.

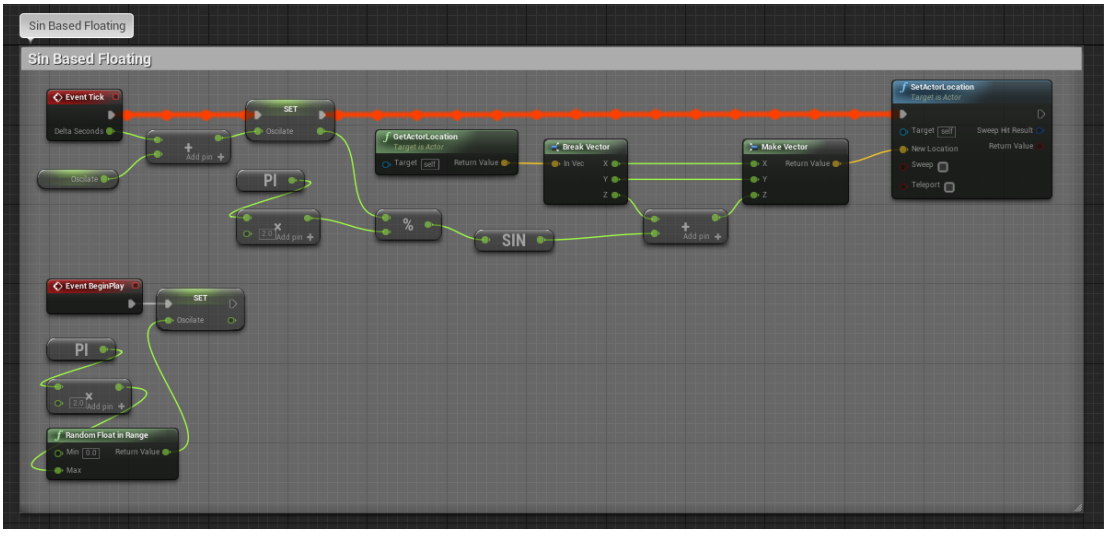
Exercise 3

This exercise laid some foundations for the creation of a prototypical first person shooter style game. The level was built up off a series of walls and obstacles to enclose a space similar to that of a shooting gallery. The player had ability to fire a simple projectile form their position and their direction which would launch a sphere in that direction until it either collided with a wall or with a target element. The game ended when all targets had been hit. The character was created to move in cardinal directions and with the ability to look around, which was necessary for the fps mechanic. Also a hud was introduced in this lab to help visualise game play statistics.



Lab 3 - Screenshot of Projectile Mid-Flight heading towards an oscillating target.

The level was designed to direct player attention to the target end of the range, where 6 floating targets were found. Each target utilised a Sine wave and offset calculation to allow the target to oscillate up and down and when hit by a projectile would disappear incrementing a target counter. The timer element was used to provide a sense of pacing within the level and added a competitive aspect which meant multiple players could race to finish the level and beat eachothers times. The fire volumes were used to inflict pain on the player and in this level was used to restrict players getting too close to the targets.



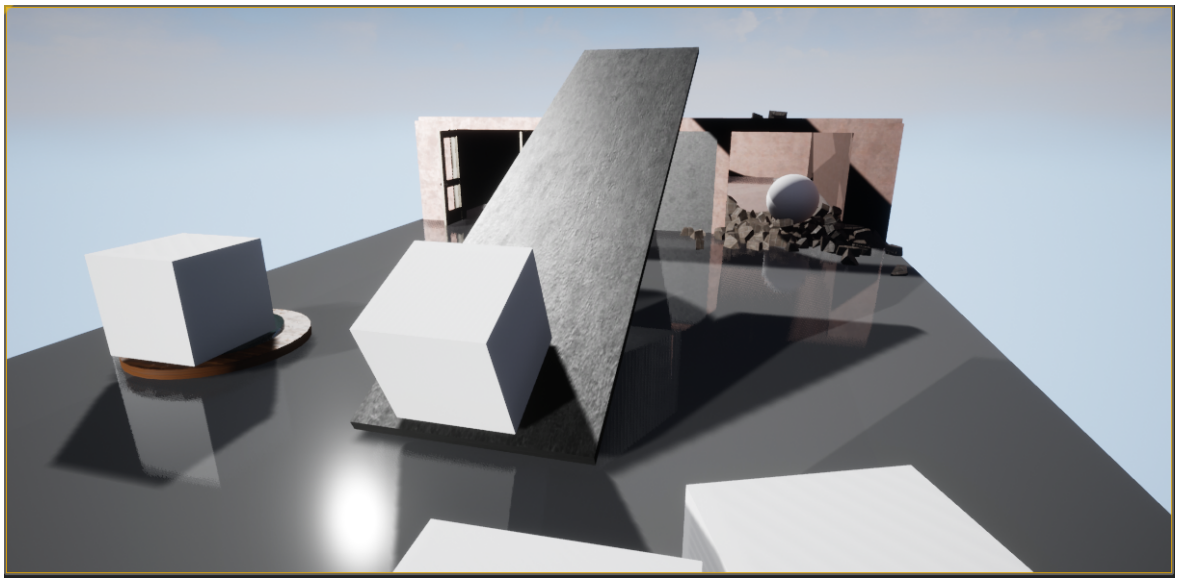
Lab 3 - Screenshot of Sin Wave Based Projectile Oscillation

Successes Although not a main part of the task, this lab was a good opportunity to play around with lighting, materials and level layout which all helped towards creating a rugged fps feel while playing. The use of a hud was introduced here and was used to display both a timer and a remaining target counter.

Failures The level could have been modified a little further to include more interesting mechanics such as a target that shot back at the player, or a different selection of weapon types such as a machine gun or grenade launcher type weapon.

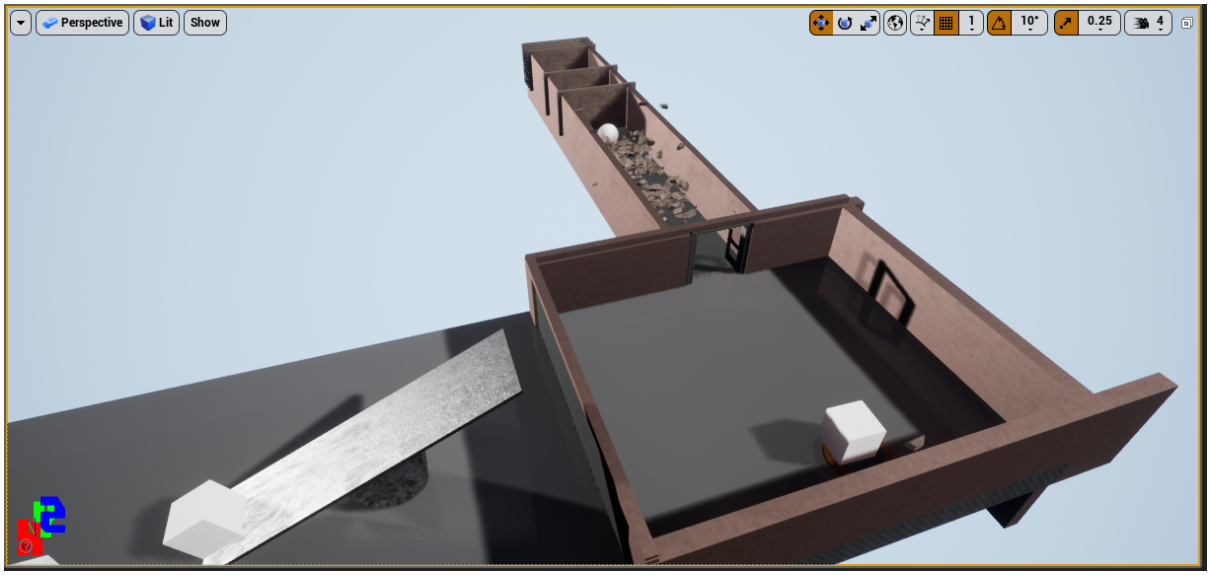
Exercise 4

This lab was built around the emulation of a Half-Life style gravity gun, which would allow for nearby objects to be picked up, dropped and launched depending on the input. The character was borrowed from the previous exercise as moving, looking and jumping was all similar requirements of this lab. The additions in this lab however was in the creation destructible elements, physics type puzzles and lateral thinking type puzzles.



Lab 4 - Screenshot of all Three Puzzle Types Aligned

The level layout was designed to force the player to work through each problem before proceeding to the next one, the seesaw puzzle for example permitted access to the button puzzle which then permitted access to the wall destruction corridor. The extended lateral thinking pattern was in forcing the player to take inventory of what they needed as once they went over the wall they could not return to fetch more items. This mean the player needed to grab cubes and cannonballs before jumping over the wall themselves.



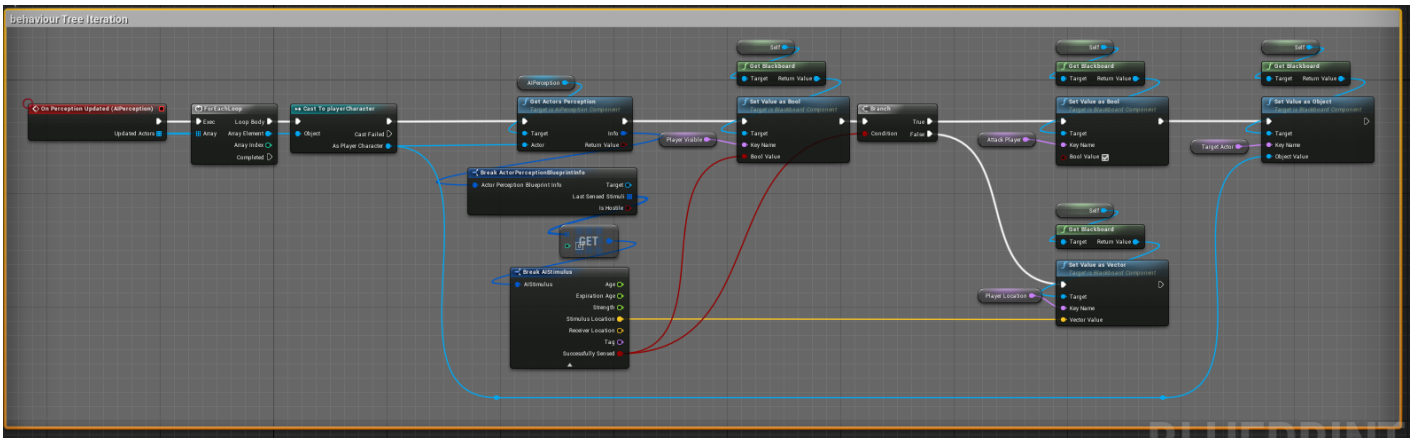
Lab 4 - Screenshot of Extended Linear Level

Successes Arranging level puzzles to force a sense of progression and forward thinking on behalf of the player and an understanding of how to build connected systems such as with the button trigger for the door.

Failures More experimentation with puzzle types could have been explored such as multiple button problems or more logic circuit type puzzles which included traps and gates. Fortunately the blueprints can be copied around freely meaning its possible given enough time.

Exercise 5

This lab taught the basics of using unreals built in behaviour tree architecture and its blackboard utility for managing and storing percept information such as player locations, or behaviour states such as whether to attack the player or not. The lab required the construction of complex blueprints to respond to perception events which in turn affect the values on the blackboard system. This ultimately drove the behaviour tree and then the task leaf nodes of the tree drove the ai controlled characters.



Lab 5 - Screenshot of Enemy AI Controller Perception Update Logic

The level was built much like in other labs except this one was built more like a maze which allowed for testing of the enemy characters path finding abilities. The addition of path finding modifier volumes allowed for a tailored search area to be laid out to the make the level more interesting. The characters had a set of waypoints which laid out a linear path but a modifier was made to the behaviour tree with a random selector which in turn held an array of waypoints which were randomly assigned to the actors pathfinding focus.

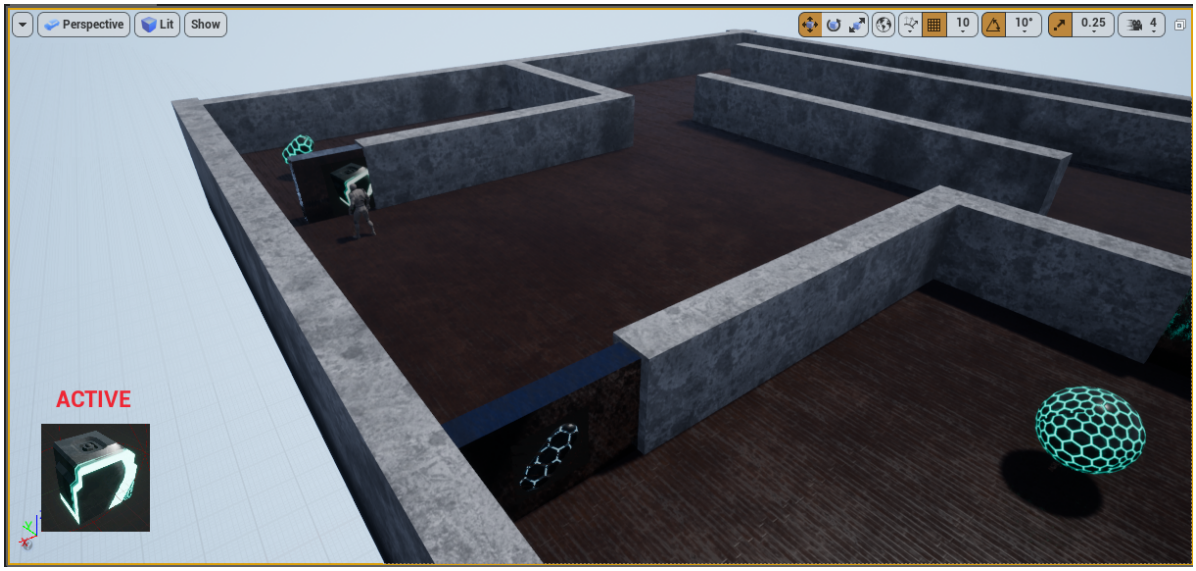
Lab 5 - Screenshot of Enemy Perception and Pathfinding To Player Character

Successes I felt initial success came with understanding this labs core lessons, such as with the behaviour tree structure, organising variables with in the blackboard and correctly modifying these variables to change the players state. The linear and random patrol behaviour made for an interesting game of hide and seek.

Failures The level was built to accommodate a test bed for building a vision dictated enemy ai, and this was tested with games of hide and seek, however given more time and confidence in the skills learned could have been used to build a survival type game.

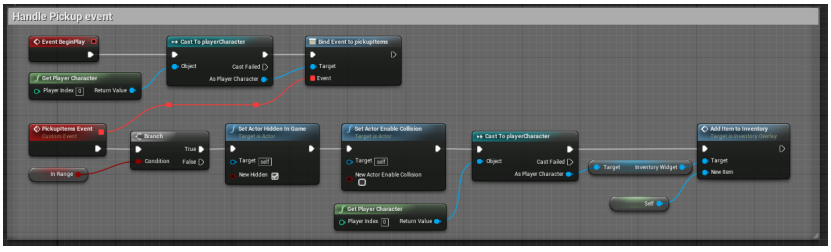
Exercise 6

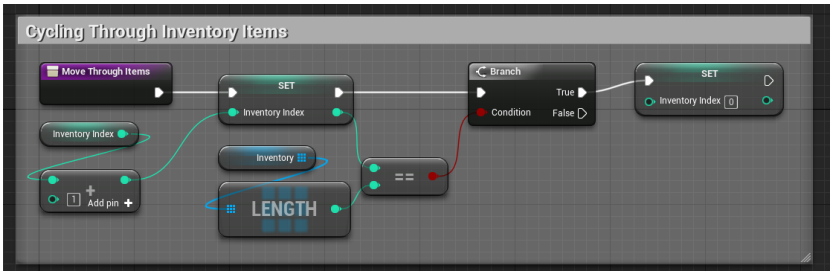
This lab exercise was a first look at building some kind of inventory system for the character player and allowed items such as blocks, pills and spheres to be approached, picked-up and stored in the characters inventory widget. The level and character were repurposed from the previous lab and modified to accommodate a more linear level progression puzzle similar to that of lab 4.



Lab 6 - Screenshot of Player Holding Primary Pickup Approaching Pickup Associated Sliding Door

The hud was proposed here to display the current active pickup which when the input “p” is pressed and within its overlap volume, will be appended to the players inventory and hidden from the level. The player can then approach a sliding door which will automatically open if the current active inventory item class matches that the door is expecting. The “e” key is used to cycles through elements in the inventory which allow the player to select the right one for the right door if its not currently available.





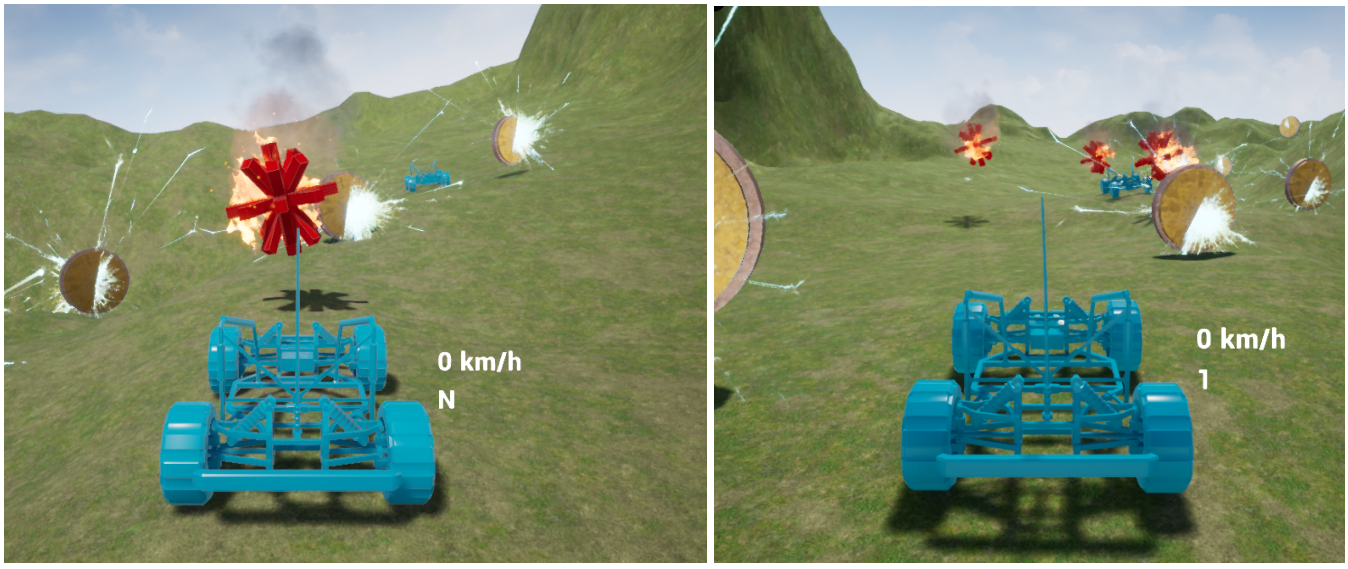
Lab 6 - Screenshot of Blueprint for Pickup Event and Inventory Cycle Feature

Successes Learning how to tie player characters to inventory items and then learning how to subtype items for unified functionality within unreal opens up many potential styles of game that can be built using this system. The inventory items acting as keys also added a nice extension on the puzzle like concepts from lab 4 and the level design from lab 5 demonstrated how to combine mechanic concepts together

Failures More work could have been placed in designing interesting items and maybe building better puzzle systems which again may be a bi-product of focusing too heavily on the technicality of the labs and not on the underlying mechanic being taught.

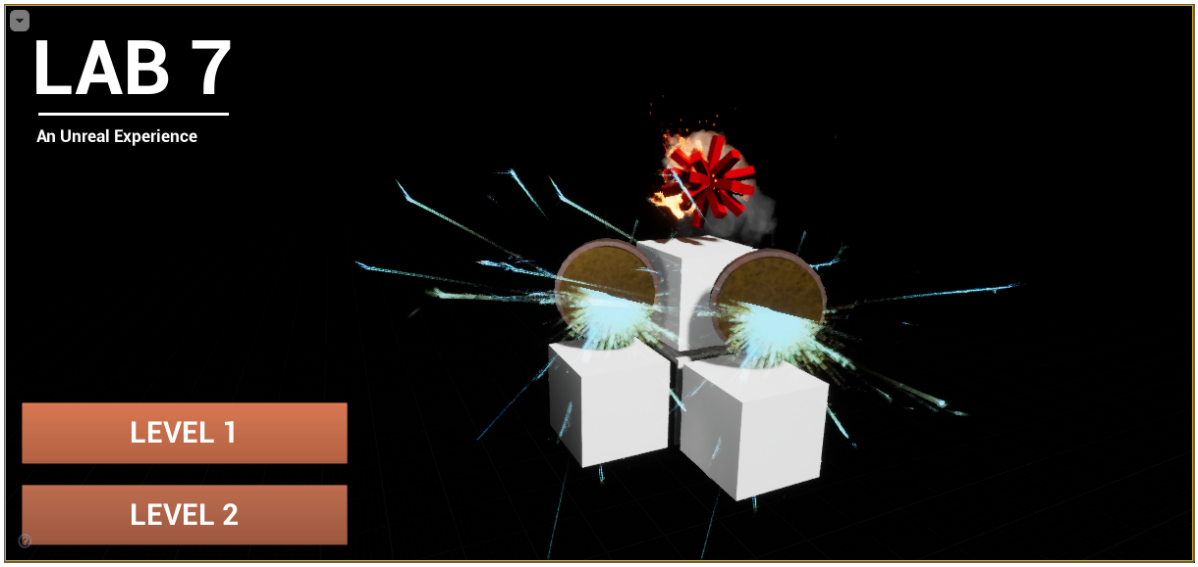
Exercise 7

This lab made use of many existing concepts from previous labs such as with objects that can be collided with with different effect or storing the state of the player such as their health and coins collected. It also introduced some new concepts such as vehicles, the possibility of multiple levels and a level selection menu at the start of the game, as well a as a small introduction to concurrent play with other players.



Lab 7 - Multiplayer Server screenshot with Damage and Collectable items in View

Now that a certain degree of comfort had been established creating both functional and renderable objects some creative license was taken with the pickup items adding both interesting materials, static meshes and even particle effects to the objects. The objects when overlapping with the car will either increment a counter integer tracking the coins collected or deduct from the health integer if collided with a damage pick up. The hud is then updated to reflect the nature of the game state. The main menu screen was created to allow for the selection of multiple levels whilst also showing a small preview of the items within the game to add visual flair.



Lab 7 - Screenshot of Game Main Menu

Successes A sense that the unreal ecosystem makes more sense and creative expression can be achieved more fluidly rather than needing to constantly refer back to the lab material or online tutorials. The Creation of the collectable items looked and worked well, the main menu system was also smooth to create and customise and the Multiplayer setup was generally without issue.

Failures Creating an interesting game mode with these components could have been better than simply getting all the items before the other person. Ideally more time dedicated to building a stronger game out of this would solve this issue.

Prototype Design and Specification

What is Electron

Electron will follow the patterns and models of a standard platformer style game but will utilise a relatively uncommon form of exploration and traversal that ideally adds its own dimension of complexity and challenge to the game. The game being proposed will make use of a simplistic model which is scalable in terms of complexity and designed to be as flexible as possible in terms of level design and addition of mechanics. The primary challenge the game is centered around is of traversing a level and reaching the end without dying. The primary aesthetic the game is of a sense of perpetual vulnerability, in that the player has no ability to retaliate to conflict or damage from enemies.

Core Game Play

The player will be quickly taught the controls which are again borrowed from the typical platformer such as the ability to move left, right and to jump. The platforming itself will make use of these controls but in terms of orbiting left and right around an object called a “Core” and the jumping will be performed by migrating between the orbits of an adjacent core. After the controls have been explained to the player through simple instruction within the intro level, the only other instruction given is to exit the level at the tunnel.

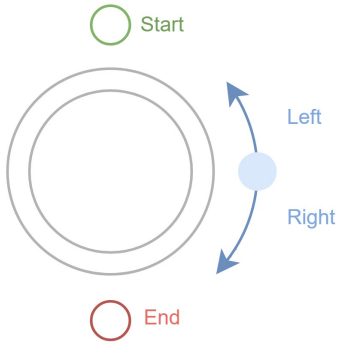


Figure 1 - Illustration of Movement Mechanic

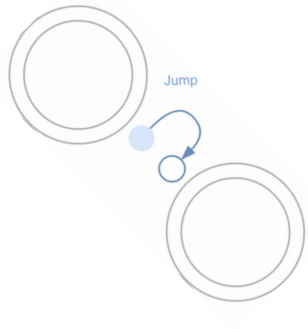


Figure 2 - Illustration of Jump Mechanic

Environmental Obstacles

The platforming mechanic in the game was designed to be a challenge in of itself in that unlike with standard platforming models such as with mario, progression is not measured in terms of distance travelled towards the right or in terms of proximity to the end but rather in terms of carefully navigating the orbits of floating circles. This means the player is force to think in terms of a rotational sense rather than in a locked horizontal axis sense.

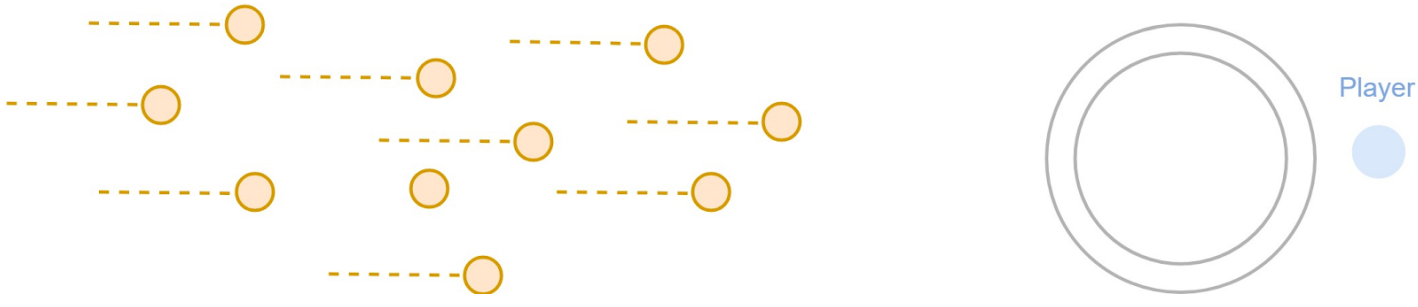


Figure 3 - Illustration of Ion Storm Objects

On top of the inherent challenge in the layout of the levels there are also environmental obstacles added such as a randomly spawned entity called an “ion storm” which creates an array of physics objects that are propelled at the player offscreen and eventually reach the player position. These objects collide with both the cores and the player and require the player to hide from the event by locating a core that is free from moving enemies and hiding on the opposite side of the core to avoid damage from the barrage of objects. The entire event is heavily telegraphed[Tel18] with flashing colours, camera shake and lines to help the player dodge incoming objects.

Characters & AI

The player character will take control of a single orbital character which is built from the same object as the parent of the enemy class, and allows for a linear movement across the circumference of the core. The player can jump between cores as well as move left and right to avoid obstacles and time progression through the level. The player also has only a single life which adds a noticeable degree of tension to the experience

After the simple “tutorial like” levels have been completed, the levels begin to include some obstacles that will attempt to slow down the player and add a secondary challenge to the platforming. The enemies in the game remain attached to their cores and follow a simple navigational behaviour such as to simply orbit around a core, to consistently teleport to an internal array of locations around the core or to bounce back and forth linearly between two locations. The behaviour is designed to be simplistic as to not overwhelm the player with information but due to the layout of the level and speed of the enemies should still serve as a competent challenge.

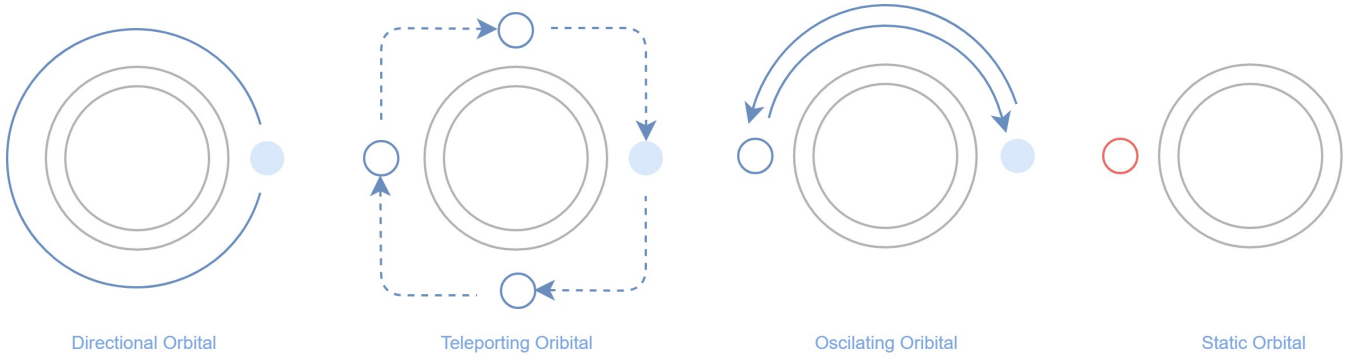


Figure 4 - Illustration of the Movement Patterns of Four Enemy Types

As the levels grow in complexity the simplistic enemy entities begin to also grow in difficulty, despite the player now having an understanding of how they work. This is due to enemy placement becoming denser, the level layout becoming larger and more complex as well as the overall faster pace of the levels. The progression ramps up the reaction time requirement and begins leaning more on the players ability to recognise patterns in the levels too.

Item Collection

Another mechanic used within the game is the addition of a simplistic inventory system whereby the player must navigate around the map and collect green diamonds which once all collected, permit access to the exit of that level. Sometimes this forces the play to take a less optimal route to reach all the collectibles or may require the player to navigate to areas of the level with more complex combinations of enemies in order to collect all the items.

Physics and Statistics

As described in the core gameplay section, the game will rely heavily on polar to cartesian model conversion to calculate the correct positions for an entity orbiting a core. The speed will work similarly to a locked axis platformer where a player applies accelerations to the player character with inputs, these inputs in turn apply to a velocity and this velocity is then added to the player position through this conversion model. This creates the more smooth ramping up and down motion of the speed and in turn adds a further control challenge to finer areas of the game. The player generally should be faster than the enemy objects as to avoid timing issues with the puzzles, putting all responsibility in the hands of the player. Collisions will make use of the in-engine separating axis collision system that can detect collisions between circles, squares and complex polygons.

Level Requirements

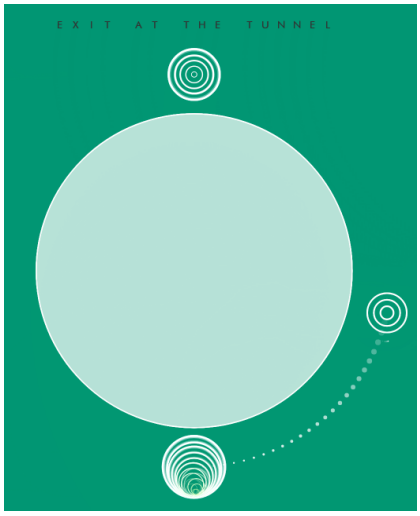
The initial levels of the game are simple tutorial like levels that introduce all the necessary information to the player slowly, and similar to portal, in a way that they may only proceed until they have understood the mechanic. The levels naturally build off of the same model of circular platforms but these grow in complexity adding more moving objects to their bodies, adding more collectible items and creating one way systems to force the player to route around the level with more consideration.

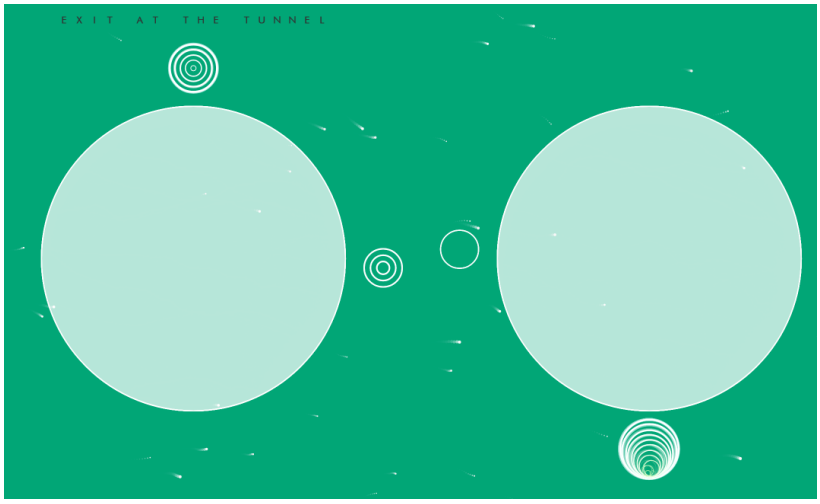
Target Difficulty

As described earlier, the level difficulty ramps up slowly at first to let the player acquaint themselves with the underlying mechanics and complexities in the game, and the later levels become harder requiring the player to combine understandings of various mechanics and enemy styles to beat the level as well as utilising pattern recognition skills.

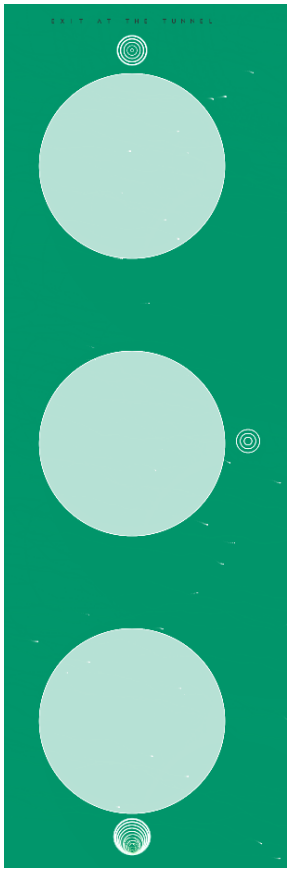
Level Overview & Walkthrough

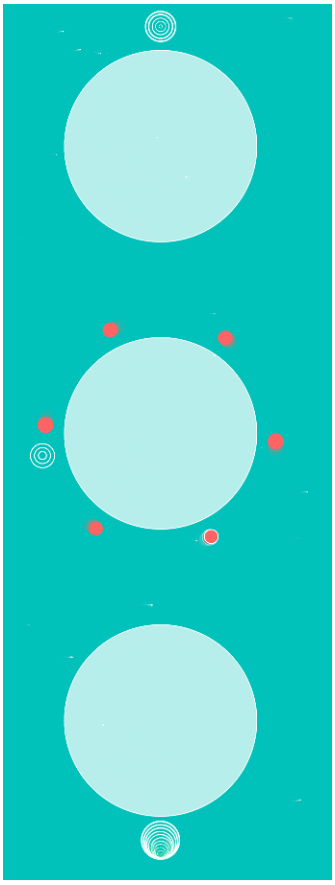
Below are a series of screenshots of each level which are in turn designed to slowly introduce and reveal to the player all the mechanics, micro and macro choices the player will need to make to complete the game.

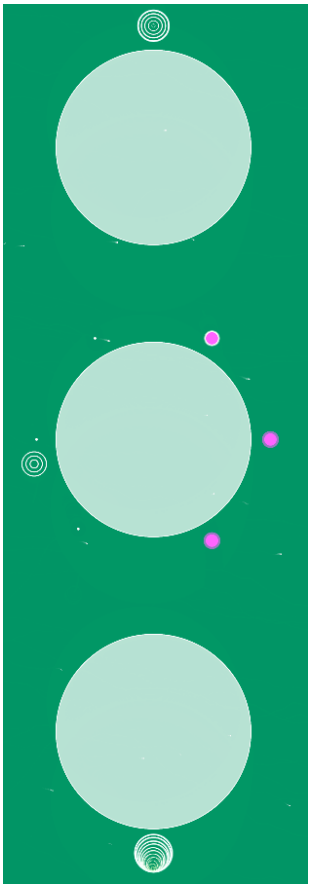


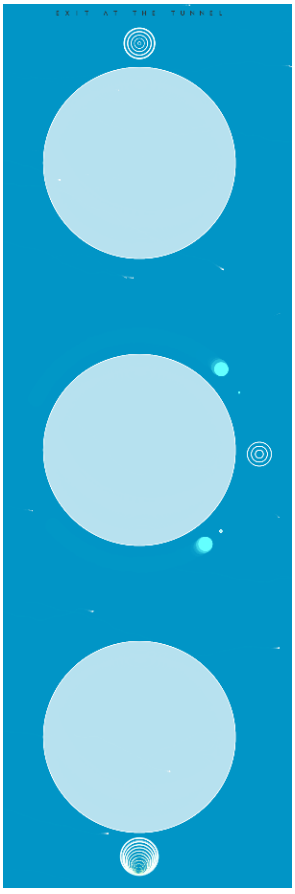


Initial Levels help to demonstrate the **movement** and **jump** mechanics where simple **visual cues** are added to help direct the player such as the **dotted line** directing the player to the exit tunnel, the **textual hints** which unlock input keys as they are taught and the **jump position** which is also illustrated on the adjacent core.

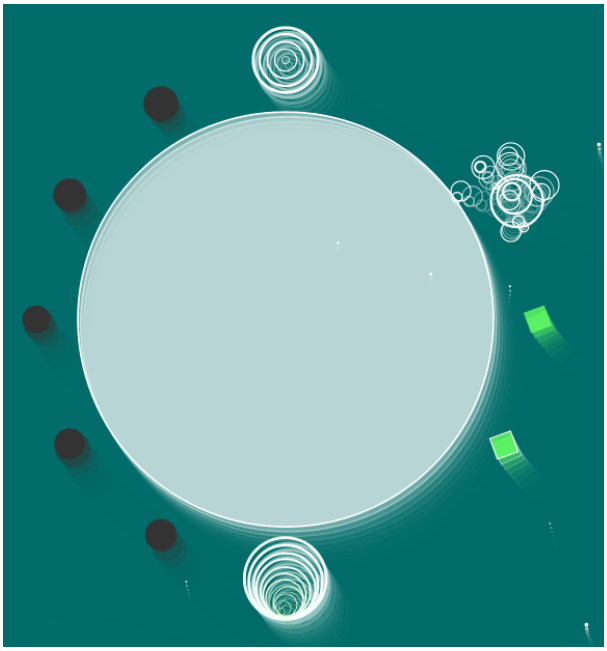


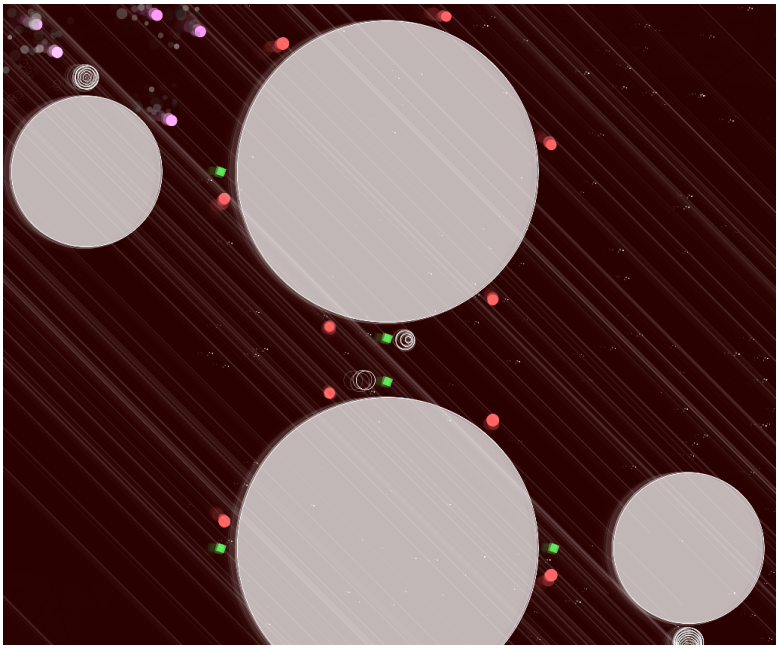




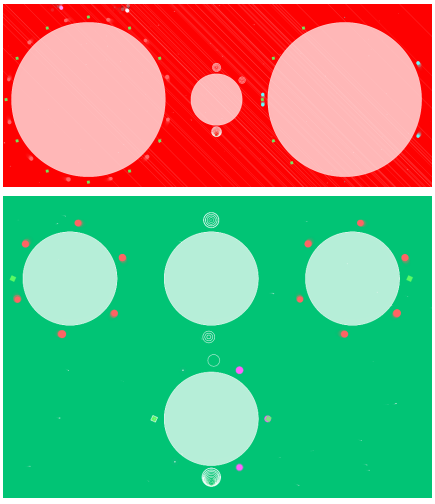


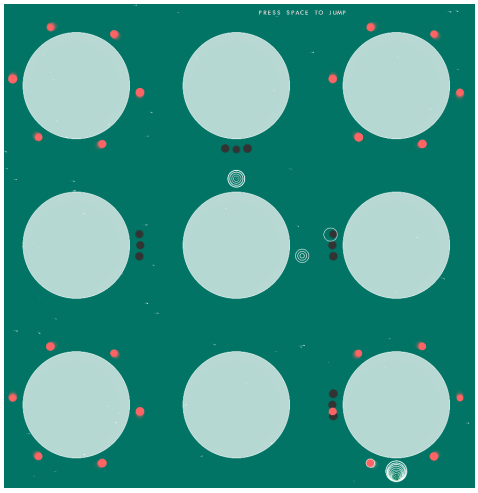
The next sequence of levels are used to help teach the player the different **enemy types** by first pushing the player through a simple downward traversal level, then introducing each **new** enemy type into the **same level** to help accommodate the player the new challenge associated with each enemy **without changing** anything else.

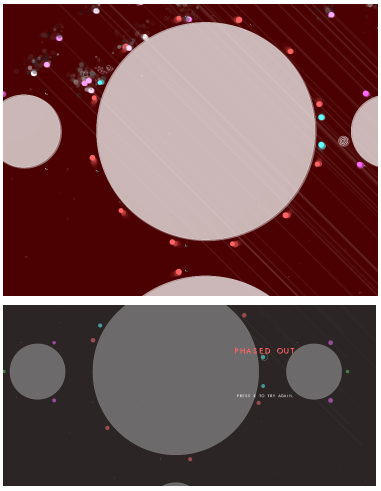




These two levels are designed introduce the idea of **collectible items** showing simple levels that the play can navigate but **forcing** them to encounter the pick up items to see their purpose. The dotted exit hint line only appears for example when **all** the collectibles have been picked up.







The final levels introduce all the mechanics in different **permutations** and **combinations** to test how well the player has embedded the platforming mechanics within their playstyle. The levels are also **longer** which increases the **chance** an Ion Storm will arrive further Increasing the **difficulty**. Some levels force the player to take a **longer route**, others force the player to time their jumps to avoid the paths of multiple intersecting enemy types.

Control Scheme

Keys	**Effect**
“A” or “←”	Will move the player anti-clockwise around the cores
“D” or “→”	Will move the player clockwise around the cores
“SPACE”	Will move the player between adjacent core orbits only if within the right angle and distance between them

Bibliography & Inspiration

[Tel18] Mike Stout, Enemy Attacks and Telegraphing, Gamasutra (September 2nd, 2015). [Online Article] Available; https://www.gamasutra.com/blogs/MikeStout/20150902/252736/Enemy_Attacks_and_Telegraphing.php Accessed 2018.

[Pla18] Richard Moss, 7 notable puzzle-platformers every dev should study, Gamasutra (January 17, 2018). [Online Article] Available: https://www.gamasutra.com/view/news/313062/7_notable_puzzleplatformers_every_dev_should_study.php Accessed 2018.

[Dif09] Brett Douville, Opinion: Ten Tips For Managing Difficulty In Games, Gamasutra (February 19, 2009). [Online Article] Available: https://www.gamasutra.com/view/news/113263/Opinion_Ten_Tips_For_Managing_Difficulty_In_Games.php

Resources

Sound Effects	Sonniss.com - GDC 2018 - Game Audio Bundle: https://sonniss.com/gameaudiogdc18/
Engine	Dominic Jomaa, Multi-Agent-Game-Engine: https://github.com/paperschool/Multi-Agent-Game
Font	Futurist Fixed Width Regular - Free Version: https://www.dafont.com/futurist-fixed-width.font
Trello	G54GAM - Coursework 2 - Electron: https://trello.com/b/WD3wn6JT/g54gam-electron
Git Repo	Dominic Jomaa, Electron: https://github.com/paperschool/G54GAM-CW2

Accessible Online : https://electron.dominicjomaa.com/