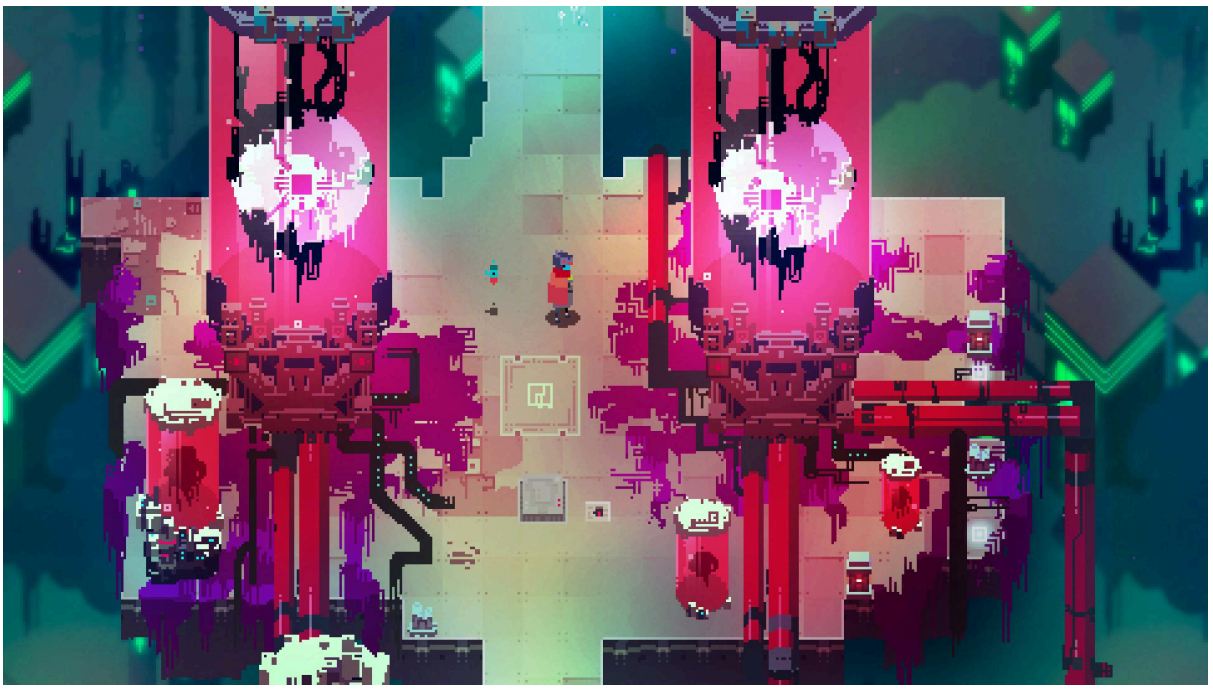
Game title: Photon Fury
Git repository: https://github.com/pjsmith97/PhotonFury
Youtube gameplay demonstration: https://www.youtube.com/watch?v=lOqDELk59wU
Group members: Nicole Levermore, Karl Clarke and Philip Smith

Initial game concept:

The player is on a 3D isometric grid map, and character movement includes a quick dash ability that kills enemies. The player is also able to shoot. The enemies shoot laser bullets in a spiral formation and move across the map, with the player dodging/shooting enemies and dashing throughout gameplay. The player has a certain amount of health, and can take some damage from laser bullets before the health bar is reduced to 0 and the game is over. The game music would be techno music with a driving beat to enhance the player experience, with a neon rainbow/ Sci-Fi colour scheme and theme. Incorporation of pixel art.

Sources of inspiration for this initial game concept:

Game: The Flow Experience by Maximilian Croissant and Madeleine Frister

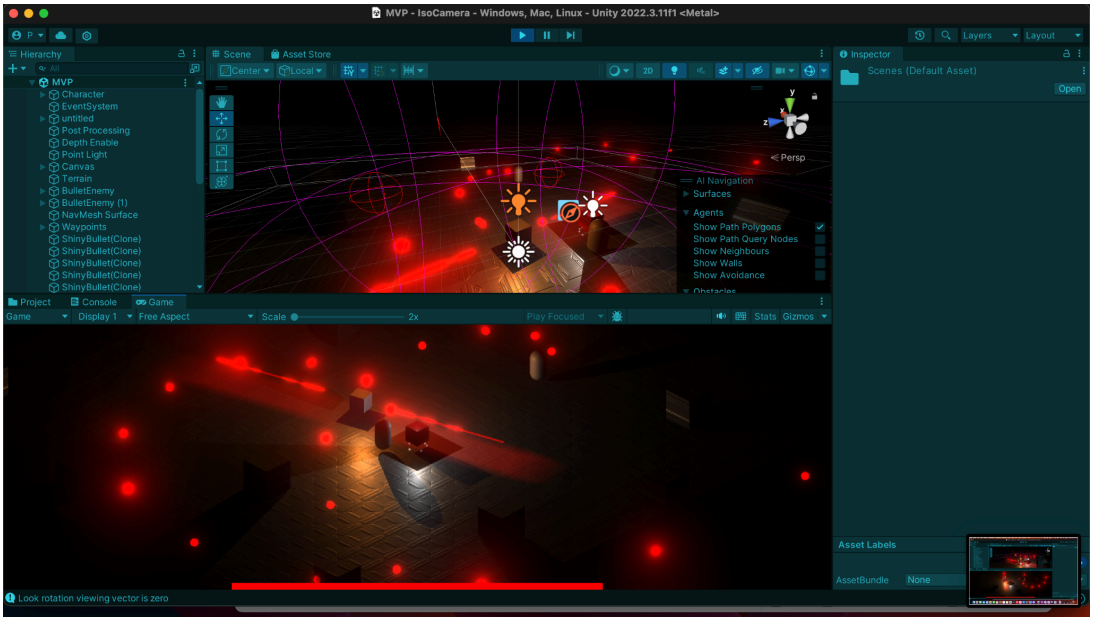


Hyper Light Drifter:

The prototyping process

We developed a paper prototype. While not significantly helpful for simulating gameplay feel due to the time sensitive and fast paced nature we had in mind for our game, the prototype did help us think about enemy variations and how they would look and interact with the player.

Screenshot of MVP in Unity Editor:

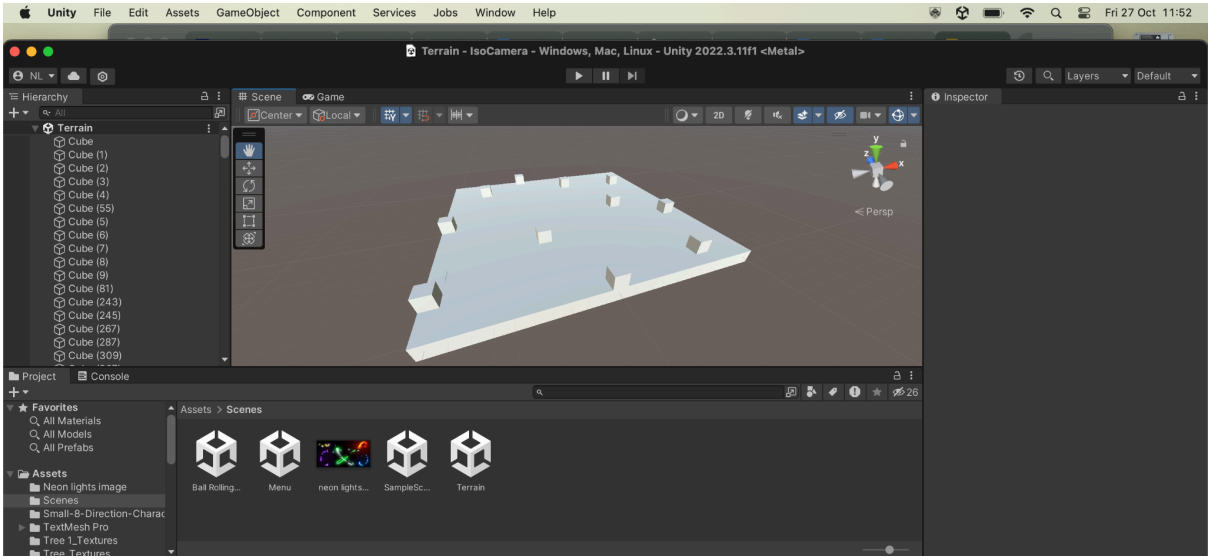


Individual development sprints

Nicole Levermore's development log: Terrain design process

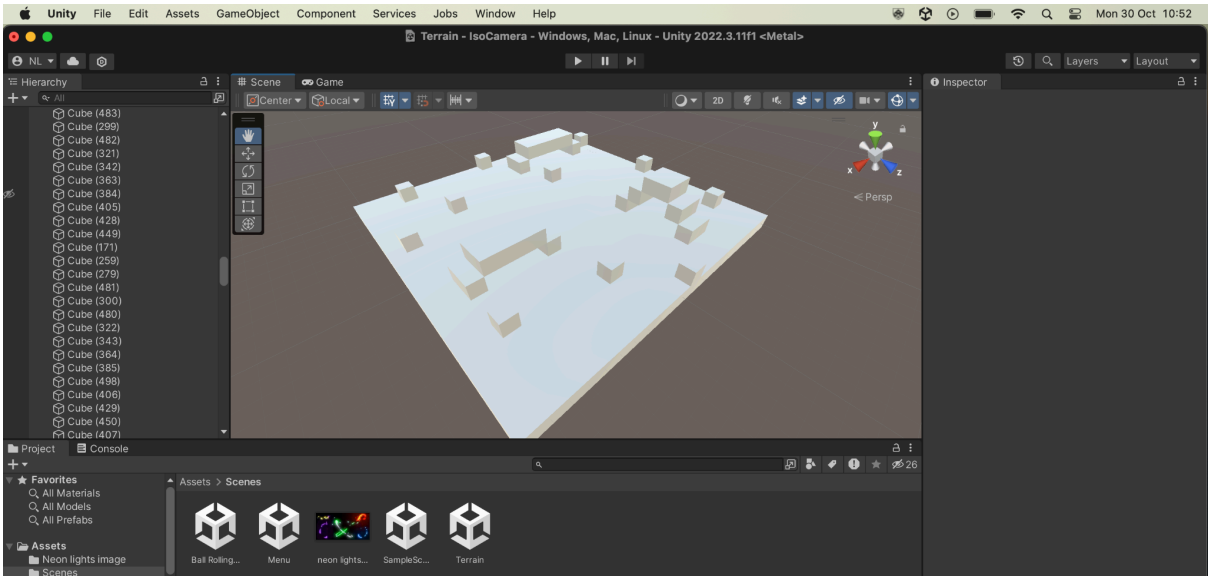
Date: 26/10/23, Time: 10 am

I used cubes to build a terrain for the game. I also added more sparsely distributed single cubes above the level plane for the purpose of providing objects for the player to hide from laser bullet fire behind and for Karl and Philip to program into the game the ability for the player to hide from laser bullet fire.



Date: 29/10/23, Time: 7pm

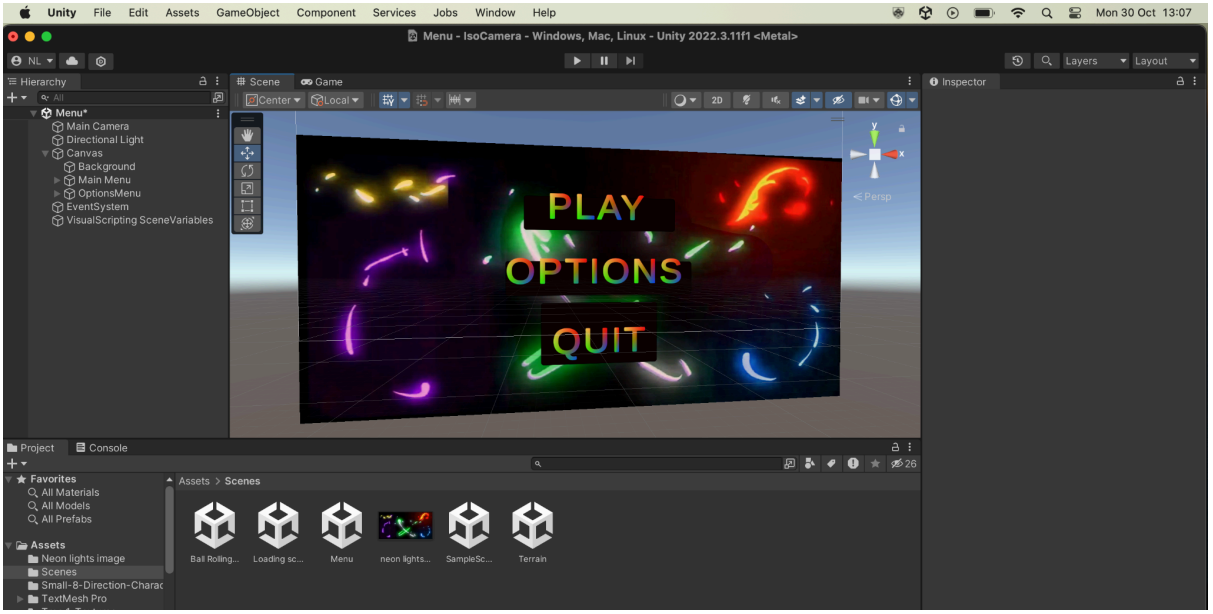
Once the ability to hide from laser bullets was incorporated into the game prototype by Philip and Karl, we decided that the level would be made interesting and dynamic by adding in points of elevation for the player to dodge laser bullets from.



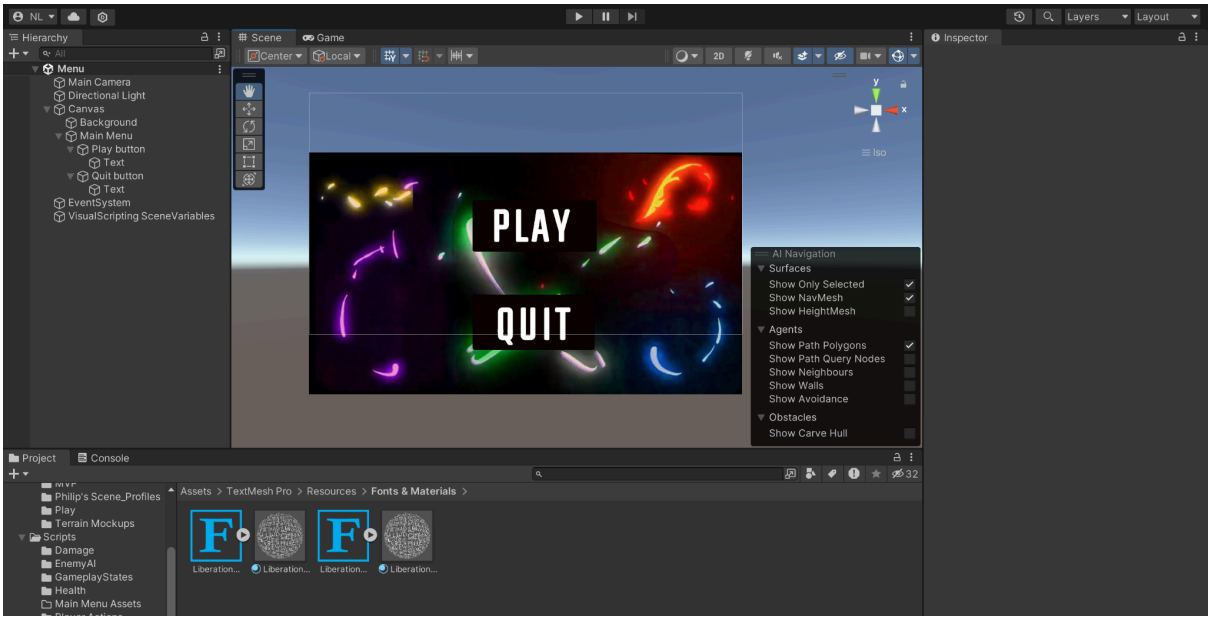
Designing the main menu screen

Date: 29/10/23, Time: 10 am

I created an interactive main menu screen for the game, supporting the ability for the player to pause and start the game. This was designed with the neon laser theme of the game in mind. The options button was placed there to eventually support the player changing the game volume, as the game will incorporate music at a later stage in development.

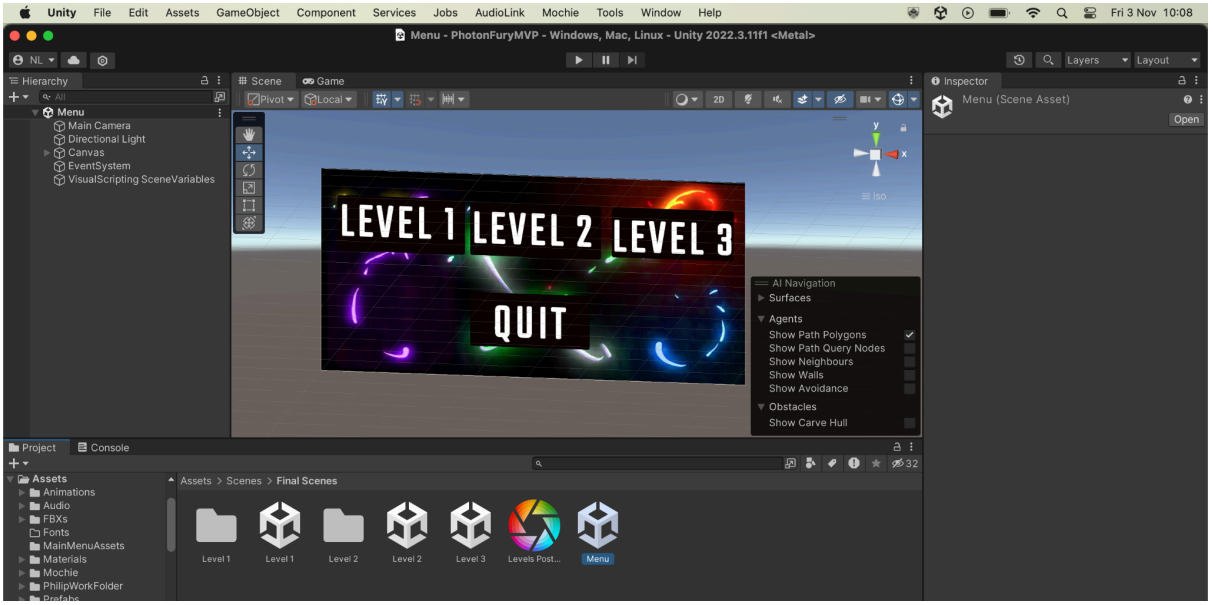


Date: 1/11/23, Time: 15:01



I changed the Main Menu, making it selectable by cursor and keyboard. Made the font ‘Teko’ and in line with the game aesthetic, using the same background as used previously. I committed and pushed these changes to our project Github. The options button was not necessary so I removed this from this latest version.

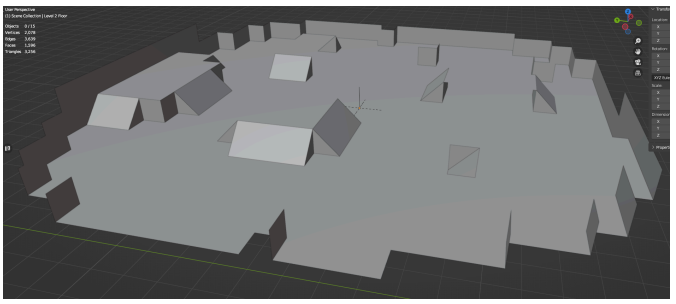
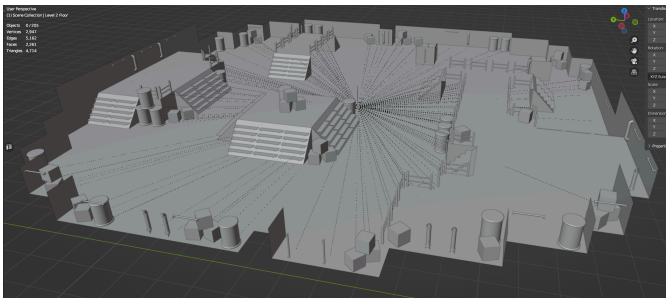
Date: 2nd November. Time: 10am

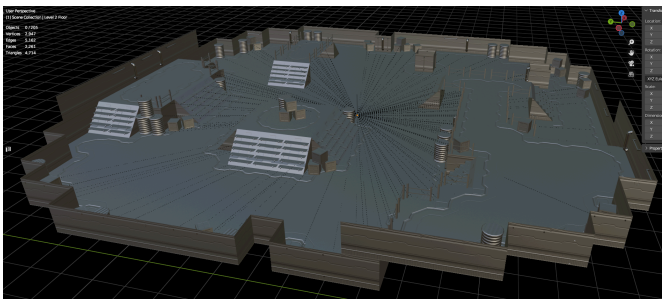
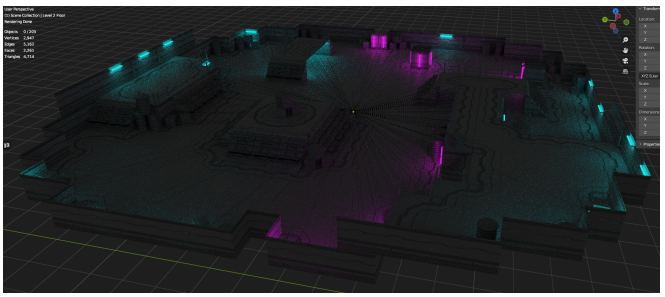


This is the final version of the main menu, with three buttons that when clicked open up the relevant level in the game.

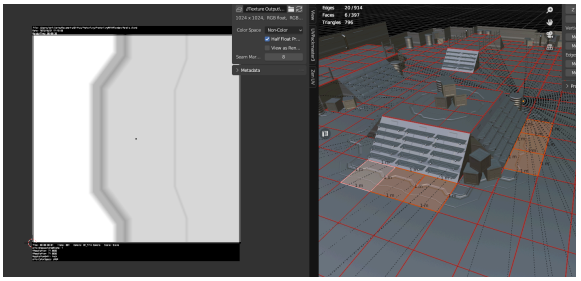
Karl Clarke's development log:

Modelling:

Base Geometry: For the level design process I took Nicole's prototype terrain and further iterated on it creating slopes and raised areas that the user could then move up on to get away from enemies. I also rounded the corners to cut down on dead uninteresting space in the level.	
Decoration: This was handled by the use of duplicated geometry. There were several smaller components that I scattered around the scene in order to add detail and decorations. Boxes, Barrels, Pipes, Cables, Neon Light Bars.	

Texture: As the textures were created by myself it was primarily a case of UV mapping the correct rotation on the floor and walls. There were many textures created for specific decorative objects that doubled up as primary textures.	
Final Render: This screenshot was captured in the Cycles renderer in Blender, giving a preview of what the scene may end up looking like.	

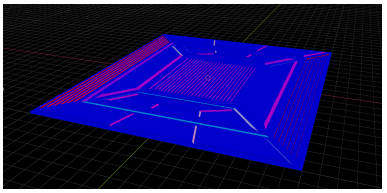
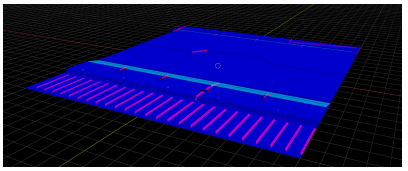
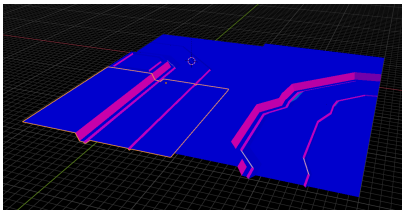
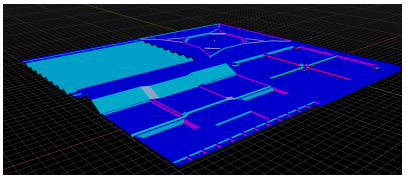
UV Mapping

This process was used for the floor and decoration textures. As I wanted a detailed floor allowing for simple patterns to be made to simulate floor patterning. I rotated each tile face to connect up with another of a similar type.	

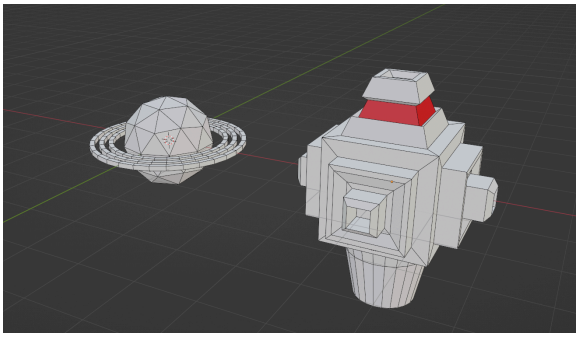
Textures

The creation of the textures was done inside of blender, using the modelling tools and an addon known as “GrabDoc” to help capture the various texture maps. The exported textures feature a Normal, Height (Bump) and Ambient Occlusion.

Displayed here are the face normals with additional information as to what these textures were used for.

Accent Panel 1: This tile was used for the stairs object in the scene.	
Wall 1: This tile was created as a wall texture to be high in detail and fit the inspiration of a Sci Fi wall.	
Floor Tiles: The floor tiles were created to be able to snap together creating a seamless shape going around the terrain.	
Detail Panels: These panels were used for miscellaneous detail.	

Character Modelling

The model (left) is the player character and the model (right) is an enemy. These were created to starkly contrast against each other for increased visibility and readability. The player character was designed after the idea of an atomic nucleus with rings surrounding it, the choice to make the primary shape be circular and round was to not appear sharp and harsh. The enemy was designed out of the idea of sharpness and cube like features. In addition to this a red eye on the top and a thruster on the bottom was created to increase positional awareness for the player. Meaning they should always be able to know where they are when on screen.	

Music & Audio:

Composition

Making the music for the game required 3 different tracks each of about 30 seconds of unique content. With the focus on ramping up the tension and energy each level I started the first level quite slow at 110 beats per minute (bpm) (typical of down tempo music), the second level at 128 bpm (typical of house and dance music genres), the final level at 140 bpm (typical of more aggressive electronic genres). All songs were sound designed in such a way as to keep to the neon aesthetic. This meant that a driving “4 on the floor” kick pattern was adhered to with use of sidechain compression allowing for a pumping and energetic feeling while keeping the main rhythm of the music clear.

Inspiration for these pieces of music were from genres such as Synthwave where primary focus is on the relationship between the kick and the pumping bass with constant 16th note (semi-quaver) pattern on the bass line. In addition to this the sound design for the bass was primarily made from hyper-saw waves that were layered to create a rich texture with an enveloped filter cutoff to add movement and an audible “strike” to the start of each bass note.

Effects & Mixing

Some notable processing was on the final piece, as this piece was more aggressive I ran an E-piano through a heavy distortion and into a small cabinet to recreate the sound of a heavy rock guitar.

In addition to this a saturator was placed after a mixbus compressor allowing for the mix to glue together better with additional harmonic content exciting the high frequencies to emphasise the guitar-like sound.

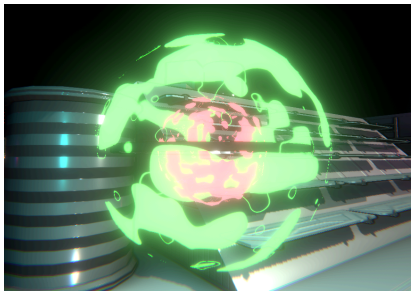
Shaders:

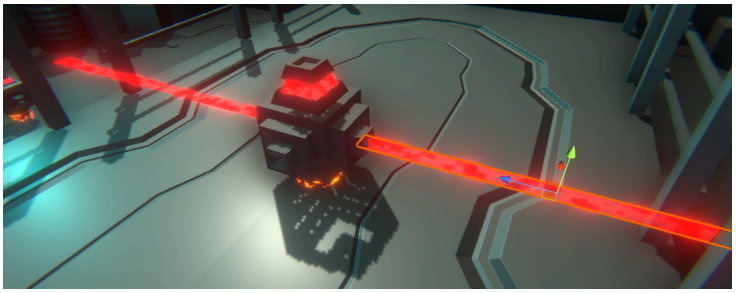
With some research I discovered that there were a few shaders that we wanted to use to add flair to our characters and game.

Shield and Lasers Shaders

For the main character and enemy we wanted an organic yet strictly Sci fi look to them. This made us come to the design decision that lasers and bubble shields should be included as part of the character design process. I found that a shader author by the name of Orels1 had recently updated their ORL Family of shaders. Primarily focused at application to social VR application VRChat, but could also be used in other projects.

The laser shader worked almost straight out of the box as it was applied over only a very small area. The shield however needed tweaking to make look correct. By default it looked too tech wise and didn't look organic enough to differentiate the character from their environment. Changing the noise texture to one using a clamped perlin noise helped to create a more organic pattern.

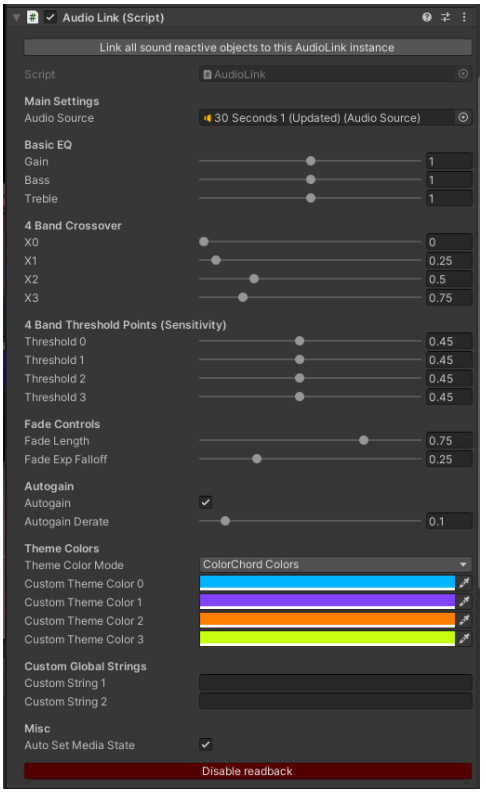




Audio Driven Shaders

The package "AudioLink" I had used before for content creation in the social VR application VRChat. However I had also seen it used as a mod for rhythm game "BeatSaber" this prompted me to test if it could work for general Unity. Appart from a few dependencies that we were not going to be using (pertaining to VRChat scripts), it functioned as expected. By using the audio input we could pipe audio in and have shaders look for a RenderTexture that the script output. The shaders could then use the pixel data to update values within the shader itself without need for a script. As this would all happen GPU side it wouldn't have a massive impact on performance.

The "Asynchronous Readback" function was also used to control the light attached to the player, a script was written to take the bass channel and lerp the value of the intensity of the light.

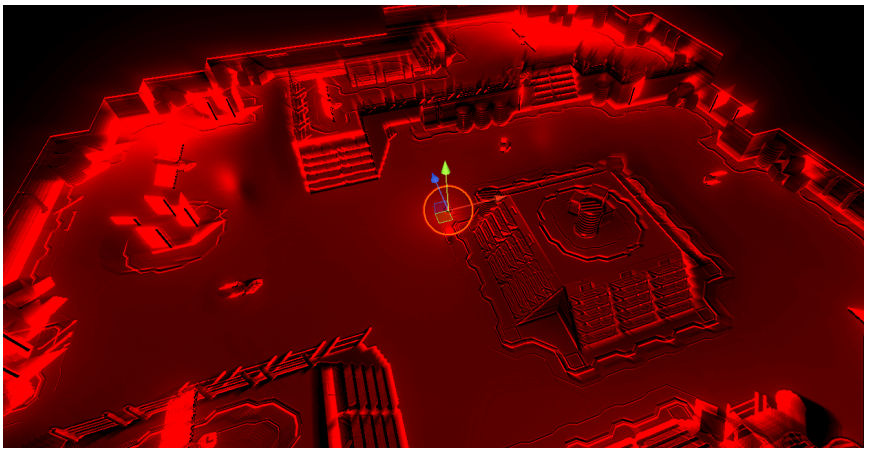


Screen Space Shaders:

Player feedback indicated that we needed a damage indicator more than just a health bar, a visual cue to tell you that you got hit and lost health. This was done with a screen space shader that I had used before from a user by the name of Mochie for its outline functionality to emphasise toon shading.

This effect utilised: filtering which tinted the screen red; shake which produced an additional image ontop of the screen and displaced it in the X & Y; and a radial blur to simulate the

impact of being hit. Together with blending type set to alpha it allowed me to blend between the raw and effected versions. Assigning this opacity parameter to an animation to trigger when you take damage finished off the effect.



Post Processing:
The post processing attached to each scene included: bloom to allow the emission to glow past SDR and into HDR range; chromatic aberration to simulate the retro feeling of an isometric game; finally, some further ambient occlusion to make the connection from geometry to geometry not feel so stark, a small amount of AO can really help make corners and harsh angles not seem so stark and aggressive.

GitHub Pushes:

October 31st 2023	
11:52	Post processing added and materials changed on laser enemy lasers
12:00	Blender & Ableton Backup
12:15	**Blend & Post Processing Fix** - Once 4 tiles to one texture for space now 1 tile per texture to avoid errors of AO across tiles
12:29	**Shield & Blend** - Created Shield material and additional floor tiles and side
12:49	**Blend Backup & Trail Width** - Trail width now set to smaller than the shield size, looks neater and less janky
13:11	**Misc Org + Animation for Shield** - Shield now grows and shrinks according to if it is enabled or disabled
13:23	Added Animation control for dash
15:05	**Dash control fix** - Detached the dash from player movement vector and changed to player input vector
15:23	**Dash and movement amended** - Switched to “GetAxisRaw” oppose to “GetAxis” so that the control feels more immediate

16:31	**Level 1 FBX** - Final layout + detail pass + UV + texture
21:54	Mochies Shaders + Texture Rework
November 1st, 2023	
00:56	**Level 2 Mockup** - Layout & Detail
09:06	**Level 3 Base Mockup** - Base geometry
11:34	**Level 3 Detail Pass & UV**
12:16	**Started work on AudioLink & Audio Controlled Lights** - included debugging the new AudioLink API which isn't 100% working with some tests
12:29	**Play Scene** - Created a testing playground for mockup of final aesthetic choices
13:34	**FBX Update** - Updated model to remove back faces as was causing issues with the mesh colliders
14:53	**New Character model + Shaders** - Extended use of shield shader to make material for character
15:09	**New enemy model**
15:18	**New enemy texture & animation** - Texture includes red light at top and thrusters at bottom, animation is bobbing up and down
15:25	**Laser enemy animation** - Added spinning to the laser enemy
20:34	**New Movement Controller** - Complete rewrite of the character controller allowing for better movement up and down slopes and better handling of the dash and jump states, now includes onSlope raycast and onGround raycast checks
November 2nd, 2023	
09:21	**Hit effect on bullet hit** - Providing players with more feedback as to get hit, mochies shaders were used to create a screenspace effect to telegraph that the player got hit
10:43	**Animation trigger to health script** - Animation moved from bullet hit to player take damage
11:00	**Added font & finalised folder structure** - Added Akira font for menus and created final folder for amassing scenes
12:14	**Added movement script to final level 1** - New movement script to test prefab in environment

13:45	**Finished Level 1 + New Prefabs** - New prefabs with models and new scripts to allow for easy implementation
14:27	**Finished Level 2** - New prefabs used to make this one quickly and easily, small snags with script dependencies not coming across but easy fix manually
14:47	**Finished Level 3** - New prefabs used to make quickly
14:42	**Enemy death particle effect** - Used unity's particle system to create an "explosion" of voxel cubes with the thruster texture to look like an enemy explosion, enabled physics for interaction with the game world
18:09	**Pause menu + win condition** - Canvas for retry and go back to main menu, used Philip's win condition to add "You win" to the screen upon winning the level

Philip Smith's development log:

October 26th, 2023

Made bullet enemies for bullet hell gameplay.

Followed online tutorial: https://youtu.be/YNJM7rWbbxY?si=ky0481ACZmeNT-BD

October 27th, 2023

Used Karl's logic from his laser script to apply to the bullet projectiles in order to damage the player. Implemented waypoint navmesh functionality on bullet enemies. Implemented player target navmesh functionality on laser enemies.

October 29th, 2023

Restructured health system across the game. Both Player and Enemy types use subclasses of the abstract class Health that handles health values, applying damage, and death.

October 30th, 2023 at 11:23am

Copied PhotonFury project into Github repository. Configured Github friendly Unity settings in the project file. Made Unity .gitignore file. Set up git lfs for git on my machine.

October 30th, 2023 at 2:30pm

Made scripts to now Pause and Restart the game level. Pause the game by pressing the P key. Players can restart the game by pressing the R key after they have died, killed all the enemies, or paused the game.

October 31st, 2023 at 11:40am

Reorganised some game file scripts into folders. Started redoing dashing to feel more fulfilling and seamless with a new DashAttack script. Using new trigger functionality for enemy deaths.

October 31st, 2023 at 12:13pm

Making debug log calls and changing colors of assets to see where player slows and reaches speeds to kill enemies

October 31st, 2023 at 12:31pm

Took out speed requirements for dash to make the mechanic easier to execute. In conjunction, made enemies tougher by implementing a double for the bullet enemies, so they now spawn bullets in two different directions.

October 31st, 2023 at 12:49pm

Now the dash shield and trail appears only when the player is dashing. We will probably implement an animation later to make the transition smoother.

October 31st, 2023 at 1:11pm

Setting up death for falling off the arena using transform positions. Changed the GameMenu folder to GameStates. Player now stops moving when the game is over. The game will pause when the player falls to a certain y value in order to account for possible physics bugs or if we plan to make falling off the map a mechanic of the game.

October 31st, 2023 at 3:25pm

Got rid of the GameMenu folder. Karl helped change walls to not be see-through anymore since play testers were confused as to why they could see the player character behind obstacles.

November 1st, 2023 at 11:34am

Dash now changes continuous velocity for as long as dashing is enabled. New UI for Pause and Game over menus created, but still need functionality.

November 1st, 2023 at 12:17pm

Player is now invincible while dashing to a) encourage the dashing mechanic, and b) make the shield aesthetic we gave the dash action translate into gameplay. Dashing now has a cooldown timer, completely overhauling the last speed requirements dashing had its end condition.

November 1st, 2023 at 12:18pm

Made prefabs for the Menu and Level Managers I created to handle pause, game over states, and restarts.

November 1st, 2023 at 12:21pm

Made Health Bar prefab.

November 1st, 2023 at 1:38pm

Put navmesh on the new level floor. Created y value variable to change the player's isometric perspective input. Now, if we were to ever dynamically change the camera angle, it will be easy to adjust how the code interprets player movement to maintain accurate movement relative to the player's perspective and input. Started inserting enemies into the new scene.

November 1st, 2023 at 3:01pm

Changed movement script so player input for jump is read in Update method. The DashShield variable is now referring to a GameObject to set active. Made mesh colliders and navmesh surface in Level 2 in Play Scene.

November 1st, 2023 at 5:48pm

Play Scene now is using the new player character model. I reset the navmesh surface and implemented navmesh agent AI on the enemies in Level 2.

November 2nd, 2023 at 8:15am

Made prefab of Player Character. Bullets now have a physics collider and rigidbody to detect when they hit obstacles (ie. not the player or enemy). Bullets are now destroyed when they collide with the terrain, so players can hide behind walls to avoid bullets. In the process I made a Wall Collision script written to debug when bullets and other objects were hitting the mesh collider of the walls in the scene.

November 2nd, 2023 at 11:39am

Made a UI element that indicates what button to use for dash/boost and when the cooldown is finished. It's a green circle that disappears when the player uses the dash action, and dynamically grows back to full as the cooldown progresses.

November 2nd, 2023 at 12:15pm

Implemented a label on the Health bar UI since play testers were not always aware of where the player's health was on screen.

November 2nd, 2023 at 12:52pm

Made a prefab for the new shield UI.

November 2nd, 2023 at 2:32pm

New bullet enemy models no longer have animators since they were clashing with the spin functionality of the bullet spawner script. Level 1 enemies set up is done along with navmesh functionality.

November 2nd, 2023 at 2:56pm

Another set up for the enemies for Level 2 using navmesh due to updates since last set up.

November 2nd, 2023 at 5:23pm

Created the navmesh surface for level 3 so enemies can now move. Still need to implement waypoints and fill in serialised fields

November 2nd, 2023 at 6:13pm

Implemented the Rewired Input manager package from Unity's Asset Store to give the game controller support. Will be implementing keyboard controls under this new input system next.

November 2nd, 2023 at 6:43pm

Keyboard controls now implemented through Rewired.

November 2nd, 2023 at 11:43pm

Main menu and Levels 1, 2, and 3 all have a Rewired Input Manager game object to avoid any errors involving its absence in a scene. Reduced the size of bullet colliders so the player is hit less often and more consistent with visual feedback. Player now has a higher hit box for dash to make it easier to hit enemies. Following feedback from playtesting dash speed has reduced to help with control and make the speed less punishing.

November 3rd, 2023 at 12:58am

Adjusted Dash UI to be a bit more presentable in the build version of the game. Now it's in the middle top of the screen. Lowered damage of lasers and bullets on player by half. Changed Project name in Unity Project Settings from IsoCamera to Photon Fury.

Playtesting session 1

Key points learnt about our game in the first playtesting session (30/10/23)

Playtest set up: Laptop and mouse, testing booth. Sound turned up on the laptop in order for game music to be heard by the player.

Playtester 1 feedback:

- The player was falling off of the map a lot, finding the first level of the game far too difficult to become immersed in gameplay. **Solution: Implement an easy or tutorial level for the beginning of the game to ease players into gameplay.**
- The game at this point had no main menu- the playtester highlighted this. Solution: Implement a main menu.
- The playtester felt that there was an overwhelming amount of action on screen resulting in an overall high cognitive load, and also was initially confused as to whether lasers were enemies or not. **Solution: add indication of enemies into the tutorial level.**
- Felt a lack of control as a player, character controls not responsive enough. Moreover, the dashing ability did not feel easy to use. **Solution: Further refine character controller.**

Playtester 2 feedback:

- The playtester fell off of the world map a lot, finding this frustrating.
- The playtester did not use the jump function unless prompted. **Solution: make the jump ability a key aspect of a level.**
- The playester felt they were prone to gravitating around enemies rather than moving across the map.
- The playtester found the dash ability to be unreliable, this was due to its reliance on the character to previously be moving. Thus it seemed unresponsive as it wouldn't trigger after a recent collision.
- Found timing the dash ability to destroy enemies difficult.
- Found circling around an enemy to be a frustrating process.
- Found that there was no need to move in the four cardinal directions, and instead chose to uniformly distribute the direction of the dash ability.
- Felt that a minimap would be beneficial for larger levels.
- Felt that it was difficult to know how many enemies were left at any one time point, suggesting that an enemy counter would be beneficial.
- Found that it was a satisfying experience to get the dash ability right and destroy enemies using it.
- Struggled to see what was happening in the level, suggested zooming in more in order for the player to see.
- The playtester said that the game appeared polished, and that the shaders for the lasers were 'great'.
- The playtester knew where their player was at all times and didn't have any issues with occlusion of other parts of the geometry.

Playtester 3 feedback:

- The player felt that dashing into shooting enemies didn't feel like the best strategy to use in a game. **Solution: implementation of the ability to shoot enemies to give players more choice of attack.**

Actions taken after Playtest 1:

Refining the dash ability:

It was noted that the dash ability may not be registering due to the angle at which the capsule intersects with it- i.e it is colliding into the side of the capsule. This was looked into and it was found that this wasn't in fact the issue, the primary reason for the unresponsive collider was due to a secondary collider bigger than the primary attached to each enemy.

With some players tempted to keep pressing dash repeatedly to kill enemies, a way to prevent the gameplay to be consumed by dashing movement was meditated upon. This was implemented by adding a cooldown timer to the dash ability.

Adding in Invincibility-frames, that mean the player becomes temporarily invincible when utilising the dash ability was discussed among team members and thought to be beneficial by making it so that the player does not die when using the dash ability.

Refinement of game signalling:

It was noted that often players did not feel they had been hit by laser bullets when they had been, this could be improved by adding in a screen that clearly signals to the player that they have been hit. E.g A red flashing screen, a sound that indicates pain.

Playtesting session 2

Key points learnt about our game in the second playtesting session (1/11/23)

Playtest set up: Laptop and mouse, testing booth. Sound turned up on the laptop in order for game music to be heard by the player.

Playtester 1 feedback:

- Felt that the gravity was extremely high and gave the impression of an inhibited jumping ability. The controls felt slightly unresponsive due to the high gravity setting.
- No issues with occlusion, comments made about the top of blocks not blocking the player.

- The game felt easier to complete once the playtester got used to the dashing ability.
- The playtester felt that more feedback from the game when the player gets hit would be useful, as sometimes it was unclear as to whether they had taken damage.
- Upon the end of the dash ability cooldown, the game should clearly indicate when the ability is available to use again.
- It was frustrating that the players had to die inorder to restart.

Playtester 2 feedback:

- Suggested adding mesh colliders to boxes and other features on the map.
- An indication of the player's level of health when killed would be useful to indicate death.
- Once the player loses all health, the playtester found that they could still dash and jump, so in order to prevent the playtester winning the game beyond death this should be addressed.
- A screen indication that the player won the game, as well as an indication of how many enemies are left to kill on a particular level would be helpful.
- The amount of space available for the player to move through was just enough for the player to feel agency within the game.
- Suggested a change in the controls, specifically changing to utilising the 'WSAD' keys, Shift and Space.
- The mechanic allowing for launching the character a long way was interpreted as a good thing as it made movement more rewarding and fun. They mentioned that it was a bit silly but that is a good thing and it promotes skillful play.

Actions taken after Playtest 2:

- Added 'You won' splash screen upon completion of the level.
- Added mesh colliders to boxes and other features of the map.
- Reworked the movement controller to avoid dash and jump when dead.
- Changed enemies in level 1 to introduce them slowly to the enemy types.
- Added mesh colliders to boxes, barrels and barriers but avoided doing so for smaller objects to avoid snags.

Playtesting session 3

Playtest set up: Laptop and mouse, testing booth. Sound turned up on the laptop in order for game music to be heard by the player.

Playtester 1 feedback:

- The playtester loved the visuals and music, as well as the game concept overall.

- The playetester expressed difficulty in using dash ability and that navigating the level felt punishing.
- The playtester felt that the game would be better played using a controller.

Playtester 2 feedback:

- The playtester felt that movement needed to feel better, and that a top down point of view would help this.
- Also expressed that more feedback from the game would be useful in relation to player health, suggesting that health bars being animated to provide more responsive feedback would also be advantageous.
- Suggested that more signalling from the game when the player and enemies take damage would also be necessary.

Actions taken after Playtest 3:

- Implemented controller support to give players another option besides keyboard controls
- Added explosion effect to enemies upon death to improve feedback
- Lowered the boost in speed from dashing to help with player control
- Lowered the damage from enemies to lower overall difficulty and give players more wiggle room to get comfortable with the game's mechanics.
- Increased the shield hit box of the player's dash ability to make hitting enemies easier

Assets Used:

- Blender
  - GrabDoc https://github.com/oRazeD/GrabDoc
  - ZenUV https://blendermarket.com/products/zen-uv
- Unity
  - ORL Shader Family https://shaders.orels.sh/
  - Mochie's Shaders
    - (Standard) https://github.com/MochiesCode/Mochies-Unity-Shaders/releases
    - (Advanced ScreenFX) https://www.patreon.com/mochieshaders/posts
  - AudioLink Shaders https://github.com/llealloo/vrc-udon-audio-link
  - Rewired Unity Package
    - Link: https://assetstore.unity.com/packages/tools/utilities/rewired-21676
- Youtube
  - Bullet Hell Patterns for Beginners: Coding Your First Bullet & Spawner in Unity
    - Link: https://youtu.be/YNJM7rWbbxY?si=ky0481ACZmeNT-BD