CE318 High-level Games Development

Lara Burić (1906027)

Anathema

https://essexuniversity-my.sharepoint.com/:u:/g/personal/lb19564\_essex\_ac\_uk/EQdfs-NIAsNFitrUaqanGHgBnomdto2tHaCFvKa4JsuFEQ

Features

1. Game Structure

The game consists of a main menu, an options menu that is accessed through the main menu and pause menu, a pause menu that is accessed through pressing the Escape key at any point during gameplay, and a win and lose menu. The main menu works perfectly, being able to start the game using the start button, and accessing the options menu.



Figure 1. Main menu

Once in the options menu, the user is able to press „Back“ in order to go back to the main menu, as well as toggle fullscreen mode and volume. The difficulty is a feature that exists within the option menu, but does not actually do anything. The reason for this was a lack of time due to all the other implemented features.



Figure 2. Options menu

The pause menu works within the game with some minor bugs, i.e. sometimes the buttons are not highlighted properly and not able to be clicked. The reason for this is unknown and I was unable to fix it due to the nature of the bug being somewhat random. However, most of the time the resume button does indeed work as intended. When pausing, a lot of tweaking needed to be done in order for the HUD to be deactivated, as well as the camera to stop following the mouse position. This was done using Cursor.lockState and Gameobject.SetActive(). When selecting the options menu from the pause menu, the menu appears, however pressing „Back“ in order to go back to the options menu, causes a black screen to appear. The same method was used when selecting „Back“ from the main menu options, however in this case it didn't seem to work no matter how many times I tried to tweak it.

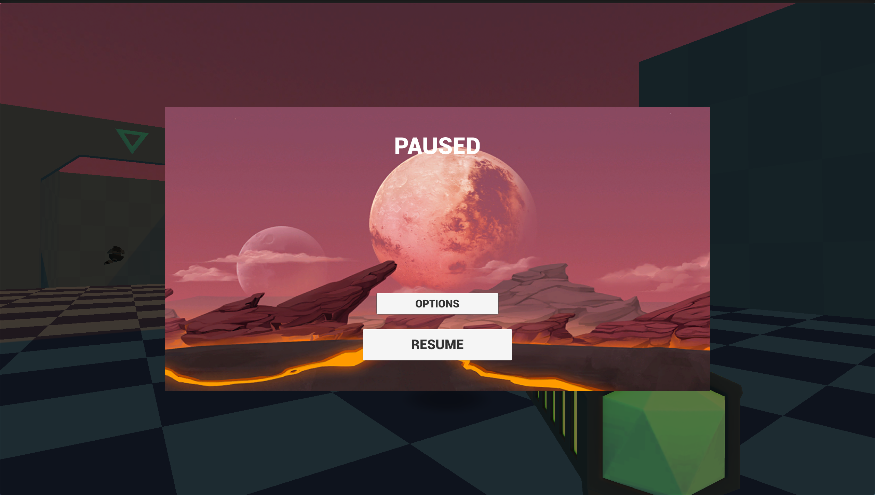


Figure 3. Pause menu

The win and lose menus work perfectly well, as well as selecting the options on them. Once the player beats the first scene, they transition into a second playable scene with a different level layout. The transition features a smooth fade in and out animation, as do most of the menus where it makes sense to do so, i.e. not the pause menu.

1. Gameplay

There is two collectibles within the game, a health pack and a different weapon in the second level which is accessed through pressing „Q“. The gameplay itself is quite smooth and enjoyable. The difficulty is rather easy, and it would have made sense to implement different difficulty levels, however, as explained before, the time constraint and complexity made it difficult to do so. The objectives are clearly set out for the player, and the „boss“ enemy provides a sufficient challenge on both levels. The new weapon on level two provides something new to the gameplay. The game features a jet pack component that is fully implementable, however for the sake of simplicity it was not added in the final version. The saving and loading feature proved to be really complex to implement, even after trying to follow different tutorials and online resources.

1. Art

The soundtrack of the game features a main menu theme, and different music for both levels which were acquired from the asset store. There are also multiple sound effects that are able to be heard throughout the game. Such as footsteps, notification sounds, pickup sounds, gun sounds, etc.

In particular, all of the main game objects have sounds associated with them; enemies, pickups, the player, UI and weapons.

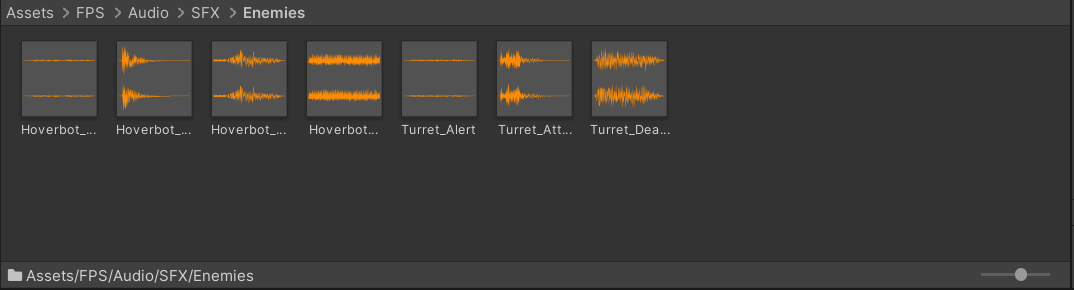


Figure 3. Enemy sound effects

The sources of light come from environment and directional lighting (imitating the sun), as well as different emissions of particles.

As the game is a FPS, the main camera is set to perspective view. This kind of game does not allow for many different cameras, however since I needed to draw the player’s gun without letting it get clipped inside the environment, I set one Camera at Depth 0 to draw the environment, and another Camera at Depth 1 to draw the weapon alone. This keeps the graphical display of the environment on the screen, but discards all information about where each object exists in 3D space. When the gun is drawn, the opaque parts will completely cover anything drawn, regardless of how close the gun is to the wall.

Multiple particle effects can be seen throughout the game, such as explosions, dust, projectiles etc.

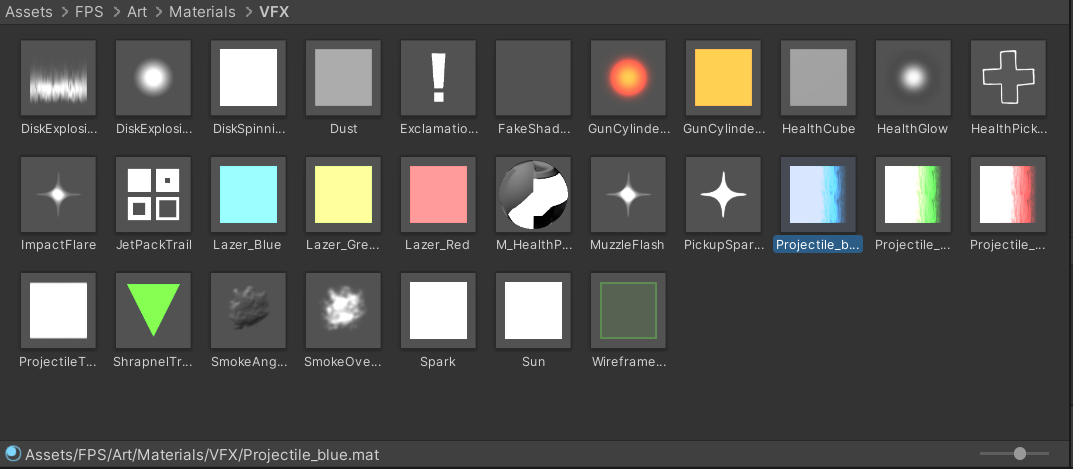


Figure 4. Examples of different paricles within the game

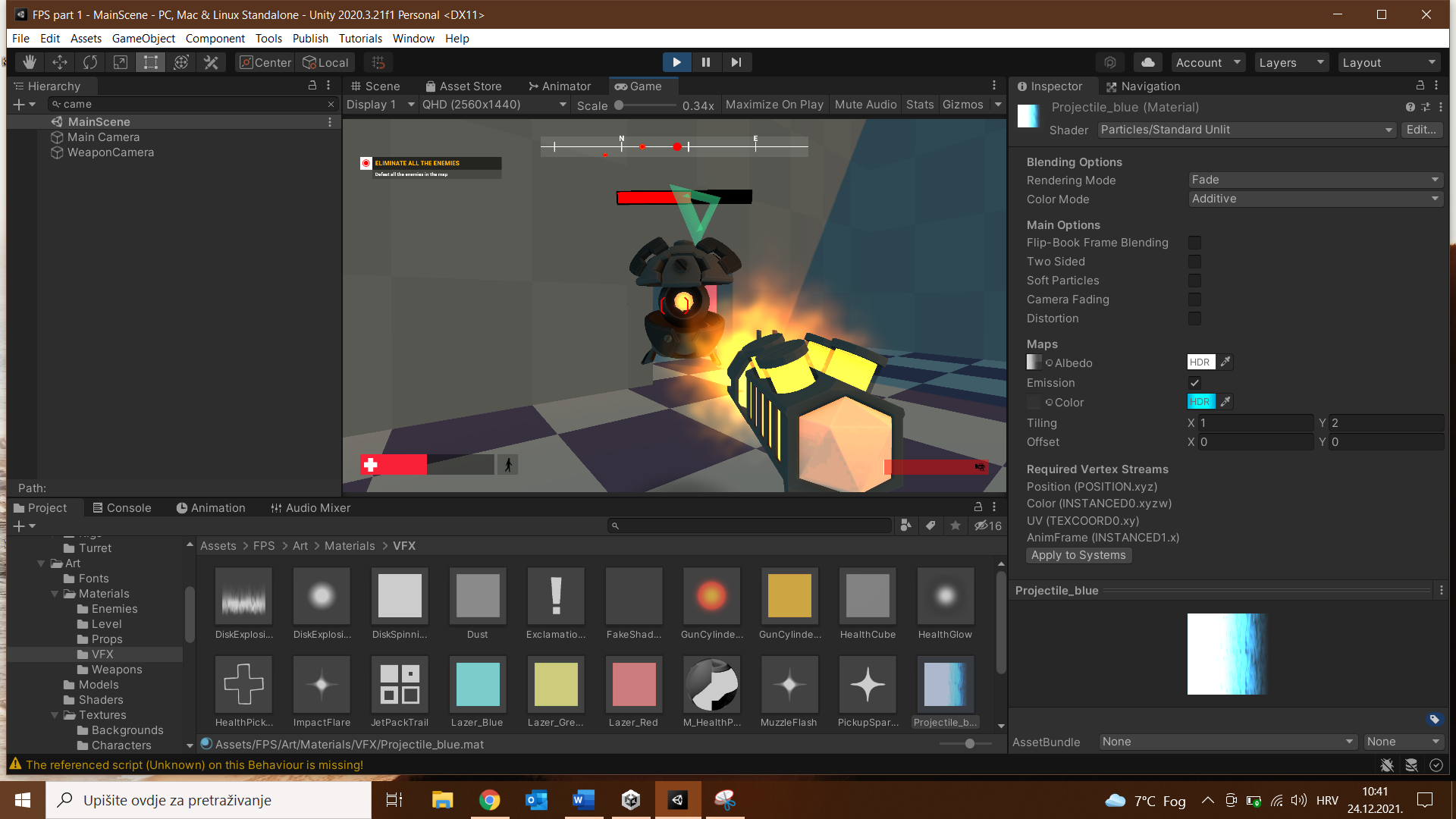


Figure 5. Within the scene, enemy and player particles and gun emission

In terms of the look of the game, many of the components use different materials, 3d models and textures in addition to a shader. All of these work well to make the game give the appropriate sci-fi look.

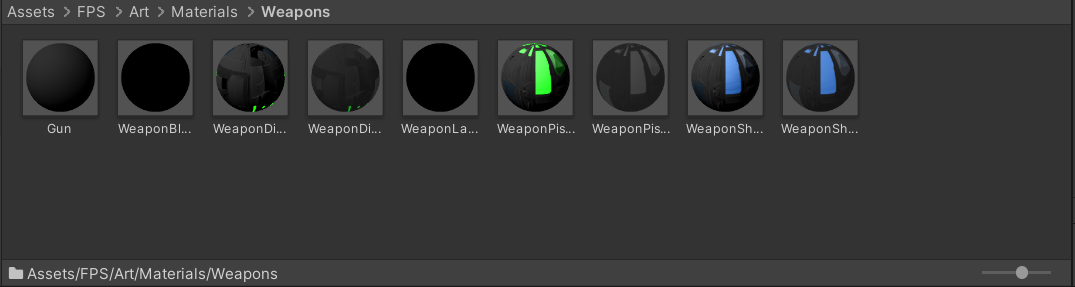


Figure 6. Different materials used for weapons

1. AI artefacts

The two AI enemies within the game are the Turret and Hoverbot. They both use NavMesh for navigating the level.

Slika na kojoj se prikazuje tekst, elektronički

Opis je automatski generiran

Figure 7. NavMesh of second level

The AI scripts are stored in the scripts folder for the game.

EnemyMobile and EnemyTurret has its own movement behavior and AI, and controls Enemy with EnemyController. EnemyController has no connection to EnemyMobile and EnemyTurret. Instead, EnemyMobile and EnemyTurret use EnemyController. EnemyController has patrolPath, another utility function that is common logic for both EnemyMobile and EnemyTurret.

EnemyMobile is an AI class for the HoverBot enemy. EnemyMobile has 3 states which is Patrol, Follow, Attack. The PatrolPath class has pathNode, meaning that the enemy will follow the paths when the state is Patrol. As soon as the enemy sees the player it will change its state.

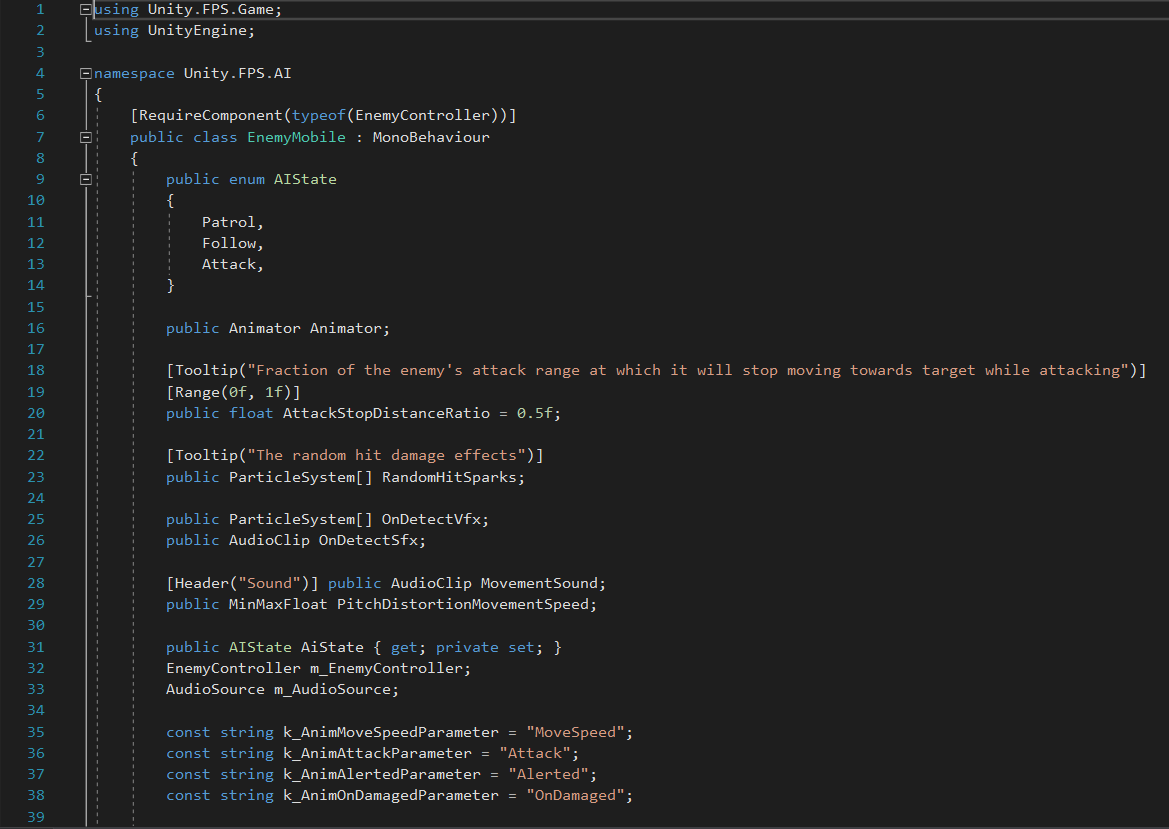


Figure 8. EnemyMobile code snippet

EnemyTurret, on the other hand, has 2 AI states. Idle and Attack.

The EnemyController script encompasses both enemies and sets up the NavMesh for both, as well as other features that they have in common.

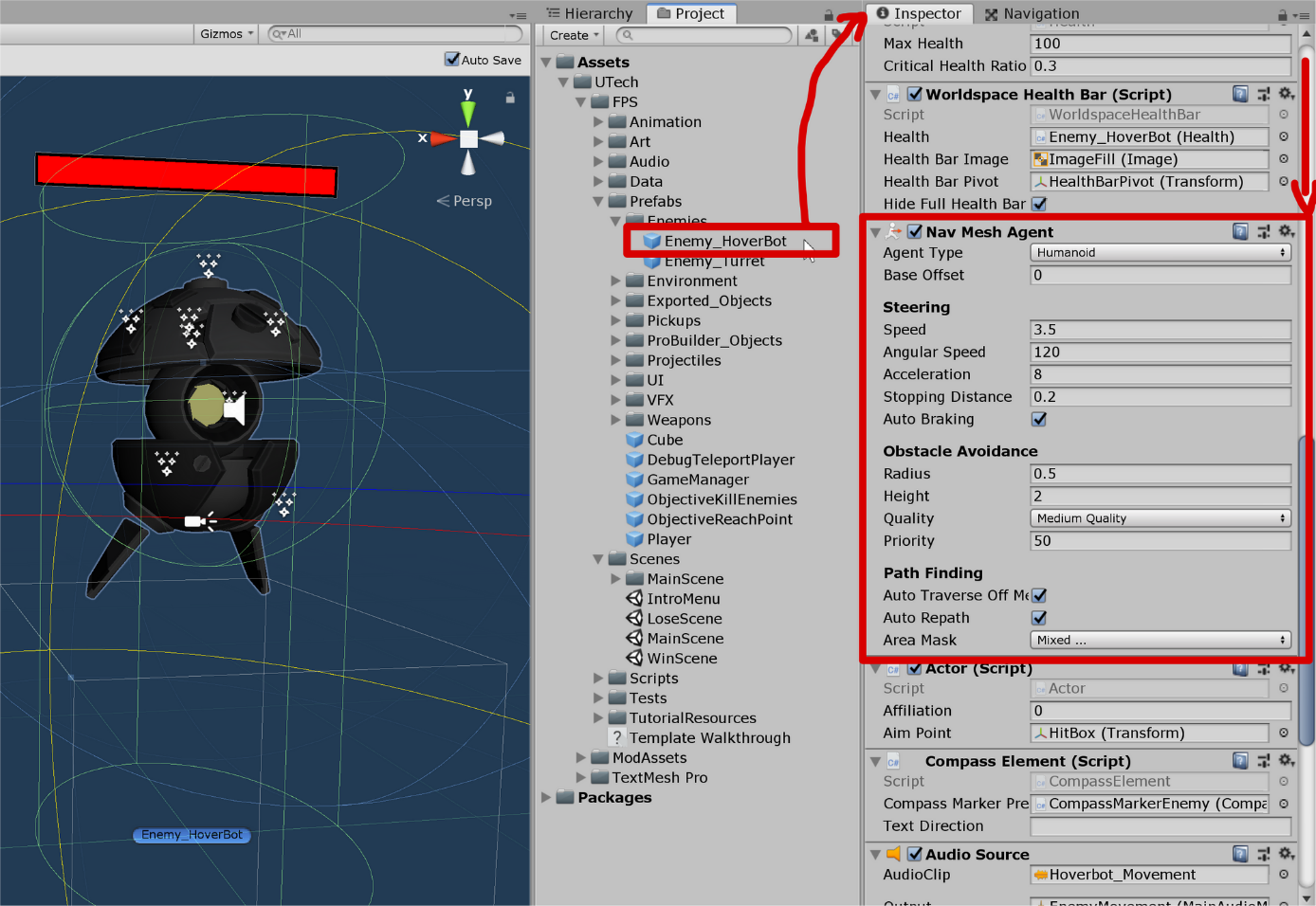


Figure 9. Closer look at HoverBot NavMesh Agent

Game Manual

The game starts from the main menu, where the player has the option to tweak some settings or start playing right away. Once the game starts, the player appears in an empty room and is able to move using the arrow or WASD keys. Once outside, there are two enemies that get alerted to the player's presence. The player is able to shoot by clicking or holding the left mouse button, aiming with the right mouse button and jumping with the space bar. After being defeated, they drop health packs, which can be picked up by simply walking over them. Once going down the stairs, the player is presented by a stronger enemy. The player is able to pick up a different weapon from the second level and switch between them using „Q“, the scroll wheel or „E“.

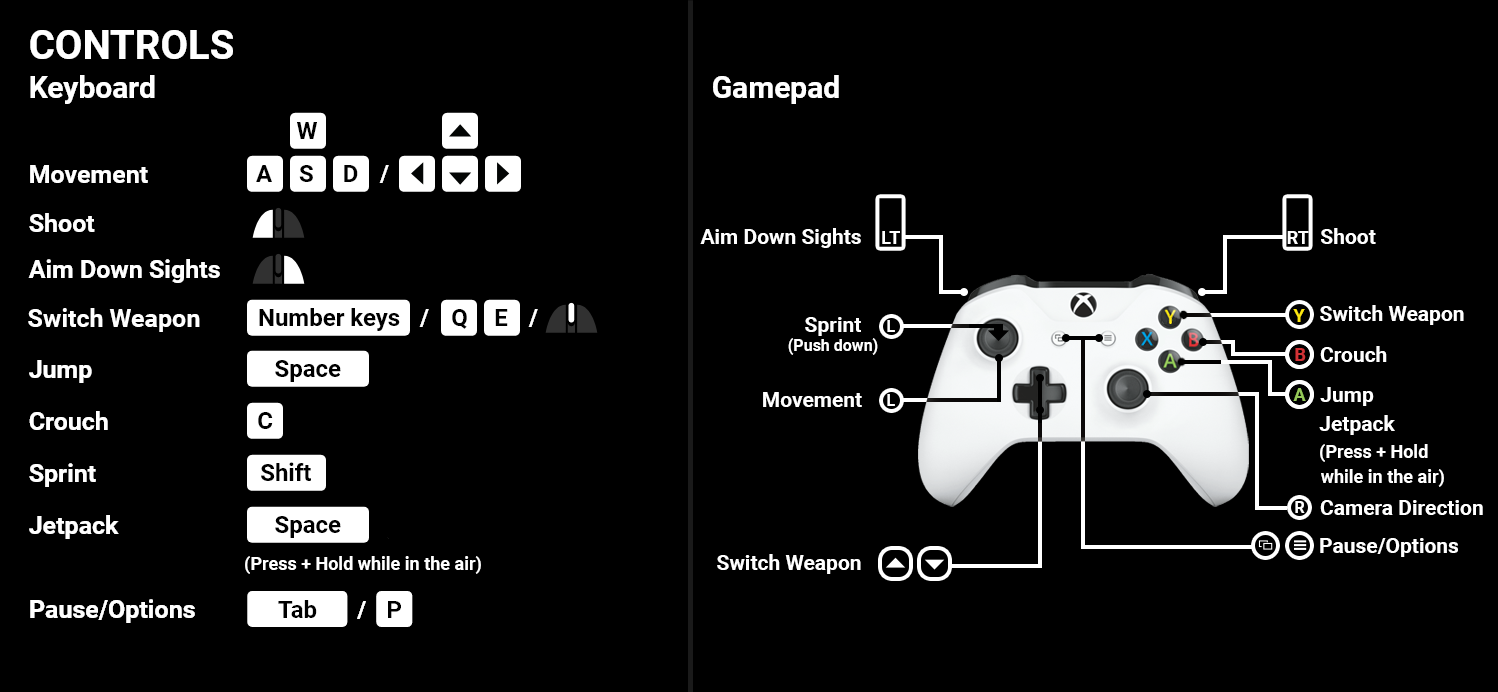


Figure 10. Control scheme

References

* FPS Microgame, by Unity Technologies (<https://assetstore.unity.com/packages/templates/fps-microgame-156015>)
* SF Sountrack Free by Mike Berns Music (<https://assetstore.unity.com/packages/audio/music/electronic/sf-soundtrack-free-187496>)
* Menu Tutorial by GamesPlusJames (<https://www.youtube.com/watch?v=BjEqZfK15Ws>)
* Settings Menu by GDTitan (<https://www.youtube.com/watch?v=iT49pNcu1jk&t=100s>)
* Various tutorials by Brackeys (<https://www.youtube.com/c/Brackeys>)
* Unity forums and other online resources