COMPUTER GRAPHICS AND 3D PROJECT

SURVIVAL VR GAME

MAIN PURPOSE

- Development of a survival game:
  - Unity 3D engine, Oculus Rift VR Technology, C# programming language
  - Environment building and modeling
  - Scripts, AI and game mechanics
  - Optimization
  - 3D game objects from Unity Asset Store
- Gameplay video:
  - In-game view + Player view
  - OBS studio + Photoshop

UNITY 3D

- A real-time development platform:
  - 3D,2D,VR and AR for games or simulation
  - Scripting API in C#
  - Drag and Drop functionalities
  - Multi-platform support
  - Many 3D features:
    - Resolution settings
    - Screen space ambient occlusion (SSAO)
    - Shadow maps
    - And more…

OCULUS RIFT VR

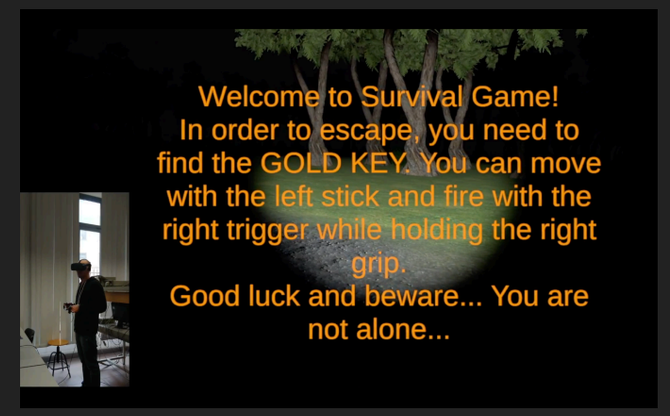
- Oculus Hardware Components:
  - Headset
  - Touch controllers
  - Tracking sensors
- Technical specification:
  - Per-eye displays with a 1080 x 1200 resolution
  - Running at 90 HZ
  - 360° positional tracking
  - Integrated Audio

C SHARP (C#)

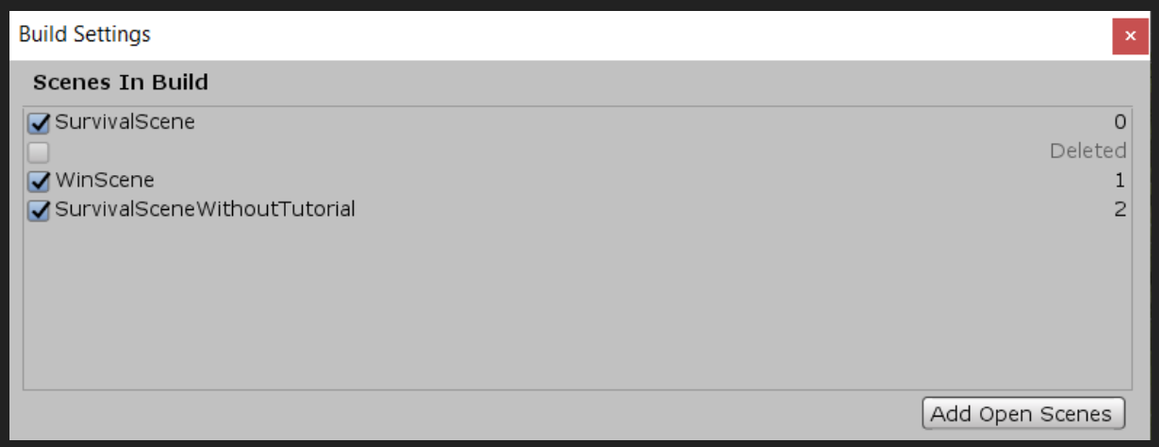
- Object oriented programming language
- Developed by Microsoft as a part of .NET framework
- Based on Delphi, C++, Java and Visual Basic

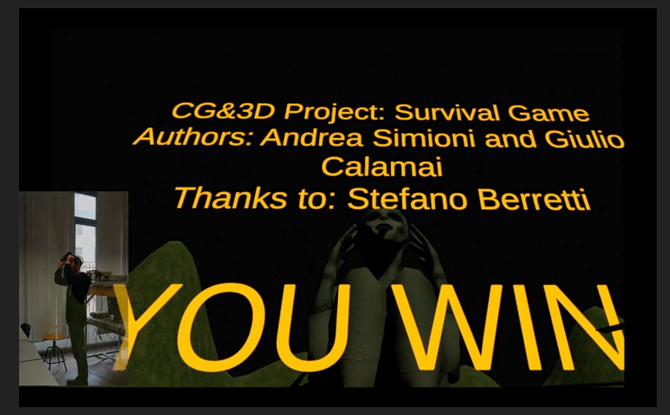
GAME SCENES STRUCTURE

- Two almost identical main scenes:
  - The first with an initial tutorial
  - The second without it
- Final victory scene



After first death



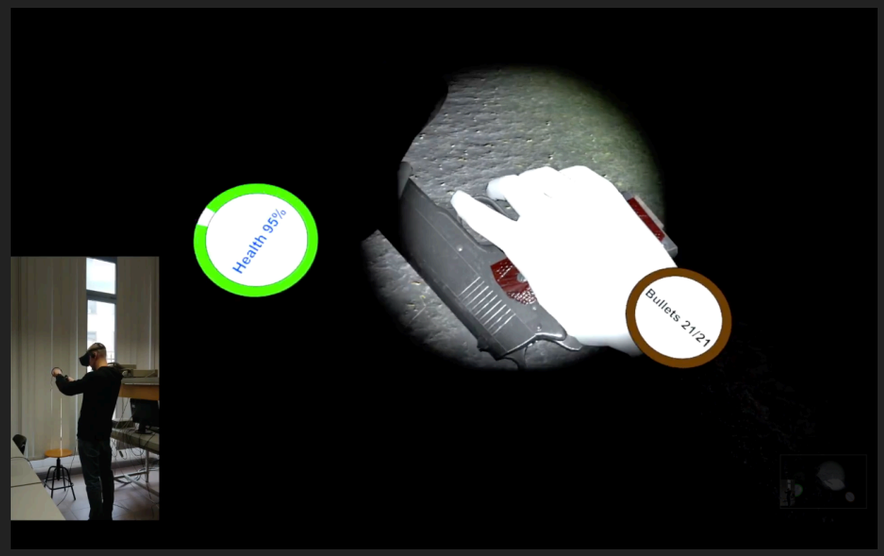


MAIN SCENES ENVIRONMENT

- A 1000 x 1000 forest area fenced by a wall
- Set in a foggy night without ambient light
- Only fire and player as pointlights
- Key points:
  - Campfire (1)
  - Escape gate (2)
  - Abandoned house (3)
  - Old village (4)

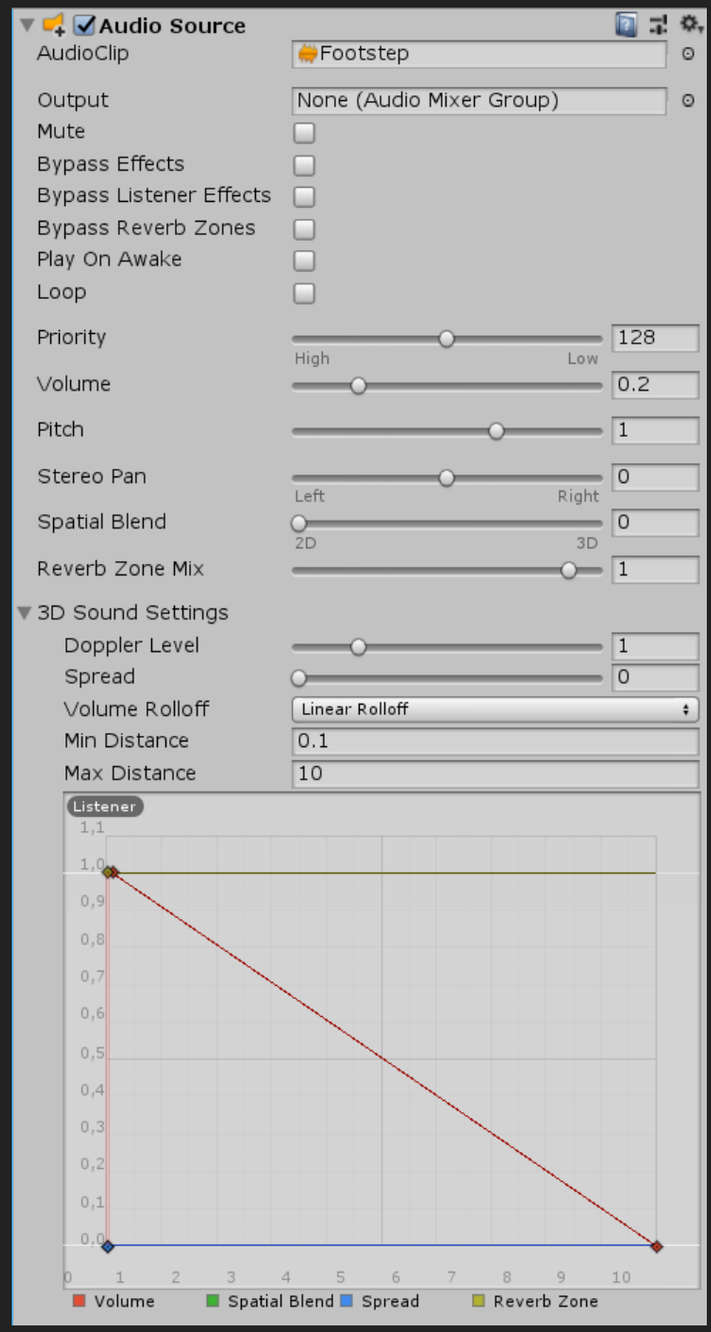
PLAYER AND ENEMY MODEL

- Player configuration:
  - Oculus SDK integration
  - Health smartwatch in left wrist
  - Bullet counter smartwatch in right wrist
  - Gun in right hand
  - Headlight
- Enemy configuration:
  - Zombie model



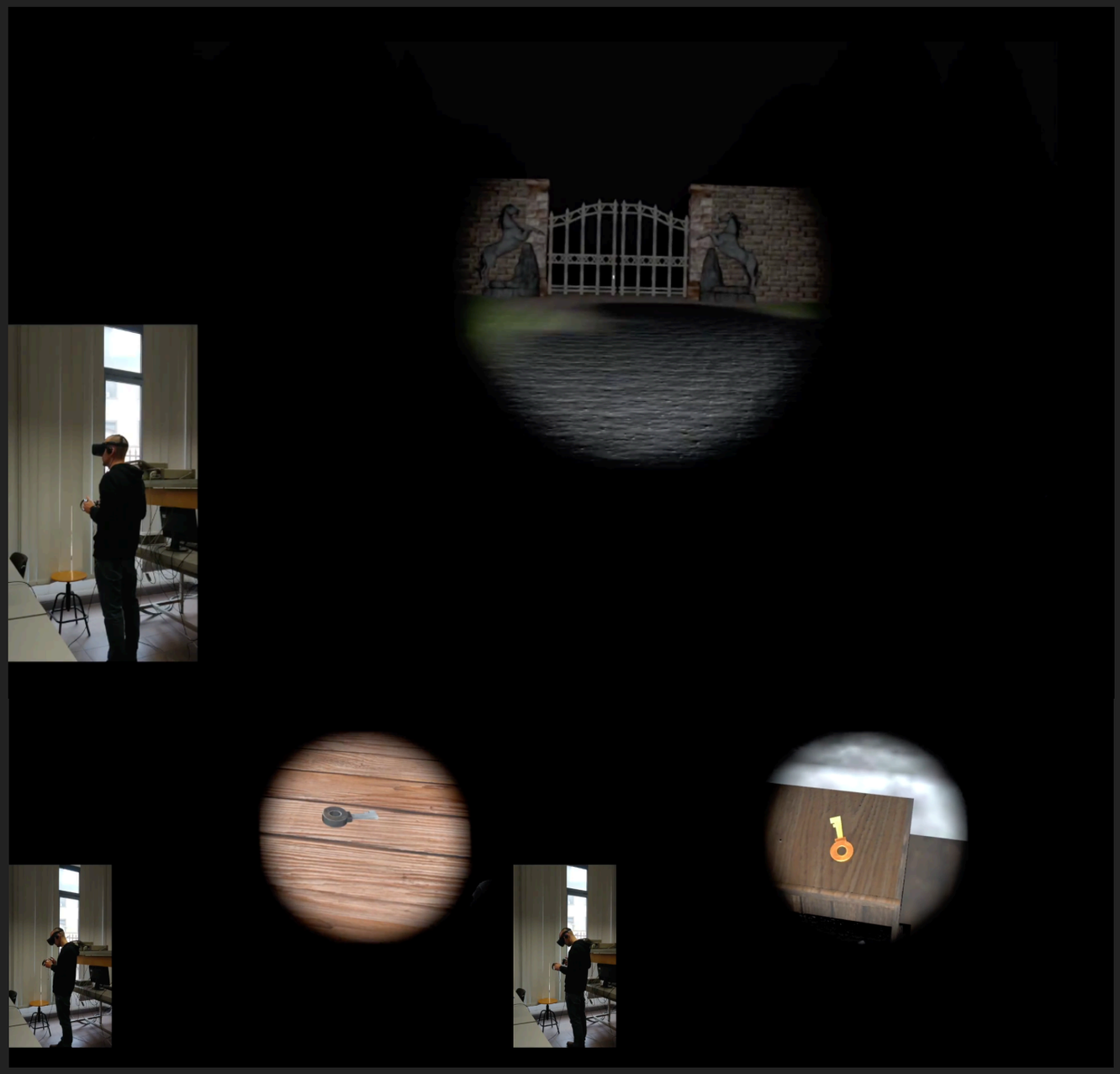
SOUNDS SETTINGS

- Environment :
  - Main background horror sound
  - lock/open gates
  - Key obtained
  - Fire
  - Cathedral’s bell
- Player:
  - Footsteps
  - Gunshot
  - Damage taken
  - Death
- Zombie:
  - Wander
  - Death



MAIN GOAL

- Escape from the fenced forest area
- Two keys to find:
  - Silver Key for cathedral in old village
  - Gold key for escape gate



CURE AND GUN RELOAD

- In all campfires:
  - Cure kit: every second 10% of player life recovered
  - Ammo kit: reload all bullets

CANVAS

- Texts and Colors for helping the player:
  - Initial Tutorial (as seen above)
  - Cure/damage
  - Need / obtained/using key
  - Death
  - Victory/credits (as seen above)

PLAYER SHOOTING

- Every time the player shoots:
  - Cast a spherecast instead bullets objects
    - If it hits enemy collider, do damage equal to 20% of his total life
    - The magazine of gun decreases by one (21 total)
- Weapon maximum range: 50 mt

PLAYER SHOOTING

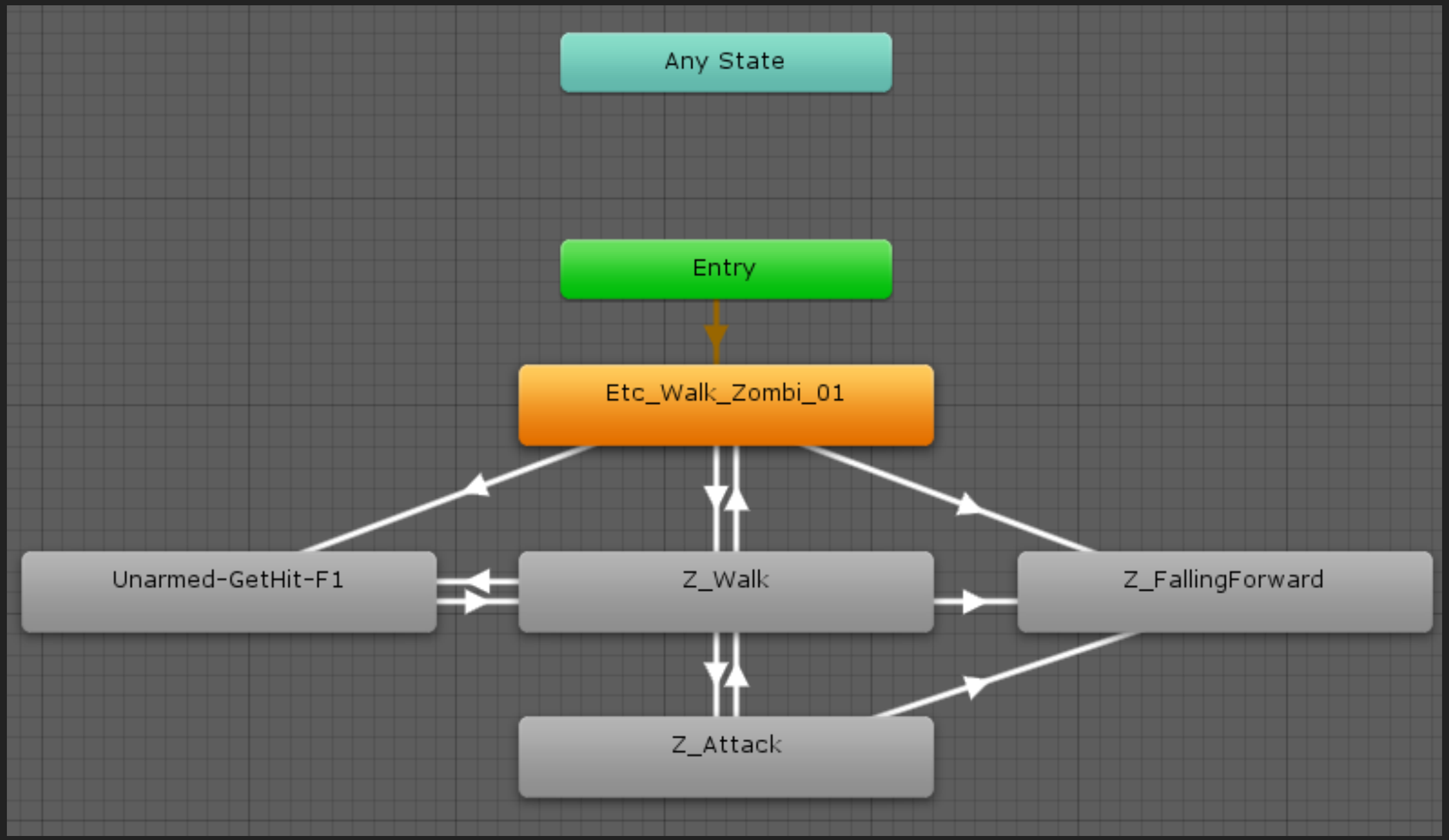
```
RaycastHit hit;
if (Physics.SphereCast(shootingAim.transform.position, 0.3f, shootingAim.transform.forward, out hit, weaponRange))
{
    // Get a reference to a health script attached to the collider we hit
    AIZombie zombie = hit.collider.GetComponent<AIZombie>();
    if (hit.collider.CompareTag("zombie"))
    {
        // If there was a health script attached
        if (zombie != null)
        {
            // Call the damage function of that script, passing in our gunDamage variable
            zombie.Damage(gunDamage);
            zombie.isHit();
        }
        zombie.OnAware();
    }
    else
    {
        // Check if the object we hit has a rigidbody attached
        if (hit.rigidbody != null)
        {
            // Add force to the rigidbody we hit, in the direction from which it was hit
            hit.rigidbody.AddForce(-hit.normal * hitForce);
        }
    }

}
```

ENEMY AI AND MOVEMENTS

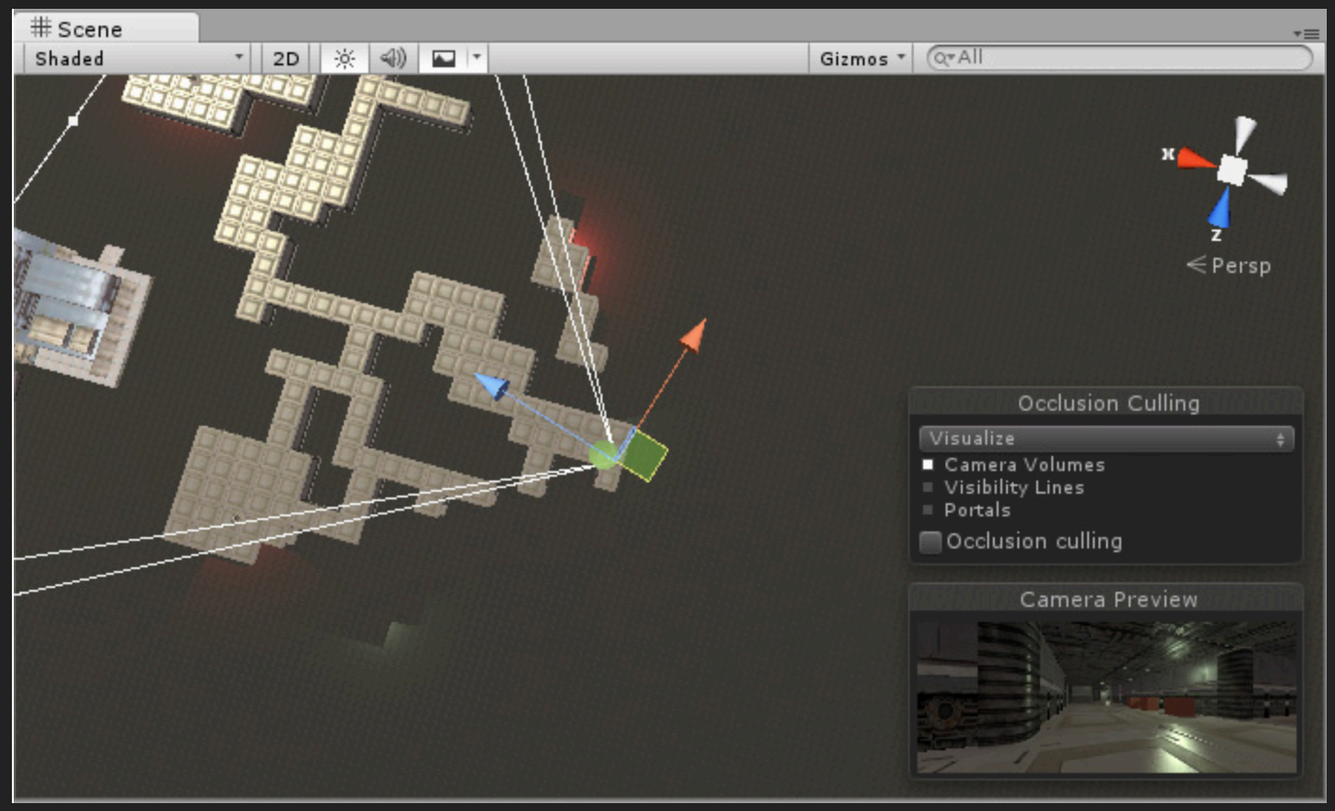
- Two movement types:
    - Random
    - Waypoints
- Update() calls always SearchForPlayer():
    - Check if the distance from player is less than 10 mt
    - Perform a linecast:
        - in case of hit with player tag, onAware() let the enemy chase the player
        - If not, continue to wander and search
    - When the distance is less than 1 mt, attack the player
    - damage equal to 15% of player total life
- After that enemy's life has dropped to 0 IsDeath() is called:
    - Death animation
    - Deactivate the enemy
    - After 25 seconds destroy the game object and re-instantiate it

ENEMY ANIMATOR



OCCLUSION CULLING

- Process which prevents Unity for performing rendering calculation for game objects that are completely hidden from view by other game objects.
- Avoid the waste of CPU and GPU computing power
- Static batching for objects that will not move, scale or rotate (environment objects)
- Dynamic batching for object that will be moving around (enemies, keys, doors etc.)



CONCLUSION AND IMPROVEMENTS

- In this project, we learned about Unity engine and techniques related, focusing in VR part
- Enjoyable and useful experience
- Future possible improvements:
  - Different environments and/or scenes
  - More enemy types with different behaviors
  - More game mechanics