Mixed-Reality Zombie Shooter for Meta Quest

Atiq Sohail Mohammed, Vishal Vijayakumar
github.com/atiq-sm/ZombieGame

Contents

1 Introduction

We built a mixed-reality VR shooter for the Meta Quest headset using Unity. In this experience, animated zombie models spawn and navigate the player's physical environment, chasing the user in real time. The player is armed with a revolver which shoots projectile bullets, to fend off incoming zombies.

Our goal was to explore real-world passthrough in VR combined with dynamic AI agents to heighten immersion and stress-testing of spatial awareness. By implementing zombies and a revolver, we aimed to create a Mixed Reality project that is both reproducible by other developers.

2 Background

Although most prior work in mixed-reality has focused on static agents, classic zombie survival titles have long explored how emergent AI, resource management, and atmosphere combine to heighten player immersion. We drew inspiration from a range of zombie survival titles, each contributing ideas that shaped our project:

- **Left 4 Dead (Valve, 2008):** We loved how the AI Director dynamically adjusts spawn rates and pacing. This inspired our spawning logic so zombies never feel too predictable.
- **Dying Light (Techland, 2015):** Its day/night cycle radically changes zombie behavior. While our demo runs in a single environment, we borrowed the idea of context-sensitive aggression when zombies chase more aggressively if you linger.
- **Days Gone (Sony Bend, 2019):** Watching huge hordes navigate obstacles taught us the importance of NavMesh pathfinding. The zombies can weave around furniture just like those massive swarms.
- **The Last of Us (Naughty Dog, 2013):** We were impressed by the intelligent AI that listens for footsteps and searches smartly. That pushed us to tune Unity's NavMesh-Modifiers so our zombies don't blindly bump into walls.
- **State of Decay (Undead Labs, 2013):** Its mix of open-world exploration and sudden ambushes inspired our anchor-based spawning, keeping players on their toes even in familiar real-world spaces.

By weaving together these ideas, we hope our zombie shooter captures the tension, pacing, and intelligent AI that make zombie survival games so compelling, which you can now experience in the real world.

3 Methods

3.1 Tools and Technologies

- **Game Engine:** Unity (version 2021.3+)
- **VR Platform:** Meta Quest headset
- **SDKs:** Meta XR All-in-One SDK; Meta XR Utility Kit
- **Rendering:** Universal Render Pipeline (URP)
- **Build Target:** Android (Quest)
- **3D Modeling:** Blender $\rightarrow$ FBX $\rightarrow$ Unity
- **Shaders:** Shader Graph (Fresnel outline)
- **Audio:** Unity AudioSource/AudioClip
- **AI Navigation:** Unity NavMesh plus NavMeshSurface & Modifier
- **Scripting:** C# in Visual Studio

3.2 Implementation

Project Setup We created a new Unity project configured for Android/Quest. Using the Meta XR SDK Project Setup Tool, we imported preconfigured camera rigs and enabled passthrough to display the real world within VR.

Zombie Models & Animation Animated zombie FBX assets were created in Blender and imported into Unity. Each zombie prefab includes an Animator component with idle, walk, and death state machines. Death animations are triggered on hit events.

Revolver Mechanics A revolver model is anchored to the controller rig. Shooting uses a projectile to detect zombie colliders. On hit, we play an impact VFX (quad with material) and trigger the Animator's death state.

AI Spawning & Navigation Upon scene start and at timed intervals, zombies spawn at random detected anchor points on real-world surfaces (via Utility Kit). We bake and build the NavMesh at runtime to allow zombies to pathfind around physical obstacles.

Game Logic C# scripts handle:

- **Zombie Spawning**: rigged to anchor surfaces and retried if spawn is invalid
- **AI Chasing**: zombies continuously set their NavMeshAgent destination to the player's current head position
- **Game Over**: triggered if any zombie reaches within a minimum distance of the player and attacks at least 15 times

4 Limitations

- **Setup Complexity:** Passthrough setup required the Meta XR Utility Kit and manual anchor placement
- **Passthrough Preview:** Not visible in Game View or Meta XR preview. Only when building the app and running it on the headset shows live camera feed
- **NavMesh Baking:** The NavMesh is baked at runtime, but the room needs to be scanned before hand and manually imported into the app
- **Bullet impact frames:** Fired bullets were too fast initially, and the collision was not being registered as the game framerate/refresh rate was not high enough. Resolved by lowering the bullet speed.
- **Model Colliders:** Zombie FBX often lacked proper capsule colliders. Required manual editing of the colliders
- **Pathfinding Errors:** Some zombies spawned out of bounds, and couldn't find a way to the player. Fixed by removing meshes from the walls.

5 Future Work

- Develop more sophisticated zombie behaviors (e.g., coordinated flanking, group tactics) that respond to the player's physical movements and cover in VR.
- Introduce diverse enemy types (e.g., fast runners that dodge shots, armored walkers with weak spots) to deepen tactical variety.
- Implement physically placed weapon pickups (e.g., revolver ammo clips and alternate firearms spawning on the floor) to promote VR interaction and resource management.
- Create a scoring and wave-based system, with in-world UI elements (floating scoreboards, wave indicators) that the player can glance at within the headset.
- Enhance visual fidelity with particle systems (blood splatter, muzzle flash) and dynamic lighting reactively attached to physical objects.