SaveZombies: AR Marker-Based Game

AR&VR Project Report

Ana Carneiro
up202008569@up.pt

José Cunha
up201905451@up.pt

Lorenzo Piarulli
up202401433@up.pt

GitHub Repository: https://github.com/lor3ny/CrowdAR

Introduction

The project was developed using the Unity 3D engine and ARCore Toolkit to ensure full compatibility with Android devices. The gameplay revolves around rescuing zombies by strategically deploying various elements, each associated with a unique marker. Selecting the correct marker is crucial, and in some cases, choosing the right orientation is equally important. In each level, there is a specific number of zombies, and to successfully complete a level, the player must save at least half of them.

The game offers multiple gameplay modes, which can be combined to create new and more challenging levels. The first mode involves removing obstacles by counteracting them with opposing elements, such as extinguishing fire with water or burning wood with fire. The second mode allows players to place bridges over hazardous areas to guide zombies safely. The third mode enables the player to deploy walls to shield zombies from incoming threats like bullets or falling debris. These modes can be mixed within the same level, adding layers of complexity and strategic depth to the gameplay.

Markers Recognition

The application implements a multi-marker recognition system, allowing users to place different markers at runtime to deploy various models. To enhance recognition accuracy, we experimented with several markers. Since achieving high precision requires markers to be highly detailed and distinctly different from one another, we opted to design our own custom markers. This approach allowed us to control the level of detail, improve final recognition accuracy, and create markers that clearly indicate their function.

Another challenge we faced was the placement of the environment. Initially, we experimented with two different technologies before developing our own solution. The first approach involved using a plane detection algorithm to recognize surfaces, combined with raycasting, which allowed the player to place the environment by tapping on the desired area of the screen. However, this method lacked the necessary accuracy. As a result, we switched to using a marker for environment placement. Since there was a risk of the marker being moved during gameplay, we added the option for players to fix the environment's position and remove the marker once it was placed, ensuring a stable and uninterrupted experience.

The game has four interactable objects and one environment that changes every level, requiring a total of five different markers.

The markers have been built using *DALL·E* prompts and then modified using *Procreate*. The markers are self-descriptive, and this is what they represent in order of the images presented on the next page: zombie environment, fire, water, bridge, wall.

Object and Obstacles Interaction

The user can interact with the environment by using various types of objects, each associated with different markers. Elements like fire, water, and wood are used to counteract specific obstacles. For example, fire will burn wood, and water will extinguish fire. The user simply needs to make these objects collide, and they will react accordingly.

Additionally, the user can spawn and position bridges to cover hazards such as gaps, acid, or other obstacles that need to be crossed. The wall, on the other hand, must be spawned, oriented, and moved throughout the level to protect the zombies from dangers like shooting turrets or falling rocks.

All interactable objects can be spawned in every level, but only a select few will be necessary to remove all obstacles efficiently. The player must choose the correct markers to avoid wasting time and risking the death of zombies. Some levels may require only one interactable, while others will demand multiple objects to solve different types of challenges.

Weather Data and Feedback

The game presents a location retrieval system that is used to do REST calls and receive weather data of the request coordinates. The weather data is provided and requested from *OpenWeather API*. Every time a level is started the location and weather routine starts too. At first, the location coordinates are provided, then, with the coordinates, a REST request is made to the OpenWeather services. The latter responds with JSON data, that is parsed to obtain what we need. OpenWeather responds with a very detailed JSON, only the generic weather definition is taken, the possible categories are: clouds, clear, snow, rain, thunderstorm, drizzle, and atmosphere (that is composed of several subcategories. We decided to regroup the categories in four final categories:

1. Rain: rain, drizzle, thunderstorm;
2. Clear;
3. Clouds: clouds and all the atmosphere categories;
4. Snow.

After retrieving the weather data and selecting the appropriate category, the user is provided with visual feedback. We have integrated a particle system to correspond with each weather category. For instance, if the real-world weather is classified as rain, the rain particle system will be activated, allowing the user to experience a realistic rain effect within the game environment.

UI

Our application features a main menu with two primary options: “Play” and “Quit”. When the user selects “Play”, they are taken to another menu where they can pick a specific level to begin the gameplay. Alternatively, if the user selects “Quit”, the application will immediately close. This layout allows users to either jump into the game or exit with minimal navigation.

The level selection menu provides additional options that make gameplay selection easy. Here, players can see buttons for each available level and selecting one of these buttons will launch gameplay at the respective level. Additionally, the menu includes a “Close” button, represented by a cross, which allows the user to close the menu they’re on and go back to the main one.

This setup makes it simple for users to jump into gameplay, go back, or exit the app at any time, providing a clear, intuitive and easy-to-navigate experience.

As for other UI elements, each level has a “Set” and a “Start” button. The goal for this is for you to be able to place the environment marker and then click on “Set” so it will not move from that place for better gameplay as explained in a previous section. This button, when clicked and after setting the environment in a fixed position, turns into a “Reset” one, where you can click to accomplish the opposite action in case you want to change the location of the environment. As for the “Start” button, you can click on it to actually start the game level.

Fig.1 Main Menu

Fig.2 Level Selection Menu

How To Play

The game can be installed only on Android devices using the corresponding APK provided. In addition, to provide weather condition retrieval, internet connection is needed.
The game can be played also without fixing the environment but is highly suggested to avoid some false recognition between different markers. To start every level is required to press the play button.

To be able to download the file to an Android phone you have to make sure that “Unknown sources” installation is enabled (this can be setup in Settings). Then transfer the file, locate it, and click on it to finish installing and executing. Make sure to accept any prompts (for example, related to camera access) that appear on the screen when first opening the game.

The game uses HTTP requests, so Android file manager will need to verify the APK before installing it.