Augmented Reality

Game Design Document

Miggiano Davide - 4840761
Morando Andrea - 4604844

Tortoise Maze Adventure

July 2024

Summary

I - Introduction

"Tortoise Maze Adventure" is an augmented reality (AR) game created with Unity and AR Foundation. Players guide a tortoise through a 3D maze using a drag-and-drop programming interface similar to Scratch. Designed to teach basic programming concepts, this interface allows players to build sequences of movements and actions to direct the tortoise to the finish line.

To start, players scan a flat surface to position the 3D maze in their physical space. They then use an intuitive UI to drag and drop command blocks, forming a movement sequence for the tortoise. The goal is to successfully navigate the tortoise through the maze, avoiding obstacles and reaching the end.

Key features:

- Augmented Reality: Place and interact with a 3D maze in the real world.
- Scratch-like Programming: Drag and drop blocks to control the tortoise’s movements.
- Sequential Execution: Commands are executed in the order they are arranged.
- Interactive Obstacles: Various obstacles and interactive elements within the maze

II - Gameplay

Upon launching the game, the user is greeted with an intuitive interface. The main screen presents the AR environment where the 3D maze is projected onto the real world through the device's camera via the plane detection of AR Foundation. Users place the maze on a flat surface, allowing them to physically move around the maze in their environment.

The interface consists of several key elements. The start block marks the beginning of the tortoise’s journey. The sidebar contains various command blocks that users can drag and drop to create a sequence of movements. These blocks are assembled on the work table, forming a coherent path for the tortoise to follow. The information text provides feedback and instructions to guide the player through the game.

Players drag command blocks from the sidebar to the work table to create a sequence of instructions. Once the sequence is complete, they press the "Execute" button to watch the tortoise follow their programmed path through the maze. This hands-on interaction helps reinforce programming concepts in a fun and engaging way.

III - Scene Structure, Components and Libraries

The game comprises several interconnected scripts and game objects that together create a seamless user experience. Below is an overview of the key components and their functions:

Game Object and Scripts

UI Manager (UIManager.cs) is responsible for managing the visibility and interaction of various UI elements, including the start block, sidebar, work table, and information text.

Direction (Direction.cs) defines the possible movements the tortoise can make, such as moving forward, turning left or right, and attacking. This script is essential for translating user commands into actions within the game.

Finish Line (FinishLine.cs) manages the behavior when the tortoise reaches the end of the maze. It triggers the activation of the next level and resets the tortoise's steps, preparing the game for new instructions from the user.

Executor (Executor.cs) is responsible for executing the sequence of steps defined by the user. It handles the logic for moving straight, rotating left or right, and attacking. This script collects the user's commands and sends them to the tortoise for execution.

Drag and Drop System (DragDrop.cs, DropPosition.cs) enables the drag-and-drop functionality for command blocks. Users can drag blocks from the sidebar and drop them onto the work table. The blocks can be repositioned and connected to form a sequence of commands.

Tortoise Handler (TortoiseHandler.cs) manages the tortoise's movements based on the executed commands. It interprets the list of directions and animates the tortoise accordingly.

Interactivity and Triggers

The game is designed to be highly interactive, with several triggers responding to the player's actions. The drag-and-drop system allows users to intuitively build command sequences. Blocks snap into place on the work table, making it easy for users to assemble their instructions.

When the tortoise reaches the finish line, a trigger activates the next level and provides visual feedback, such as lighting up a point or displaying a congratulatory message. This encourages users to progress through the levels and continue learning.

Libraries and Frameworks

The development of the Unity AR Maze Game leverages several powerful libraries and frameworks, which are integral to its functionality:

1. AR Foundation: AR Foundation is a Unity package that allows for the creation of AR applications across multiple platforms, including iOS and Android. It provides a unified API for working with AR features such as plane detection, anchors, and tracking.

2. ARCore/ARKit: ARCore (for Android) and ARKit (for iOS) are the underlying AR technologies supported by AR Foundation. These libraries handle the complex tasks of environment tracking, light estimation, and feature point detection, providing a seamless AR experience.

3. UnityEngine.UI: The UnityEngine.UI library is used to create and manage the user interface elements in the game. It provides components for buttons, panels, and text, enabling the interactive command panel and feedback system.

IV - Unity Assets

The project leverages several free Unity assets to create an engaging and visually appealing game. These assets are sourced from the Unity Asset Store and provide a variety of models, animations, and effects to enhance the game experience.

Here a list of all downloaded assets:

- RPG - Tortoise Boss Pack
- Awesome Stylized Mage Tower
- Cracked stone filled with lava
- Stylized Stones Texture

V - Future Enhancements

The game is designed to be an engaging educational tool, but there are several areas where it can be expanded and improved to provide an even richer experience. These future enhancements focus on improving the user interface, adding new gameplay features, and enhancing the educational value of the game.

Enhanced User Interface

1. Customizable UI Themes: Allowing users to customize the look and feel of the user interface could make the game more engaging. Players could choose from various themes and color schemes to personalize their experience.

2. Improved Feedback System: Adding more detailed feedback for each action taken by the tortoise could help players understand the results of their commands better. This could include visual indicators such as arrows or footprints showing the tortoise's path, as well as more informative error messages when commands fail.

Advanced Gameplay Features

1. New Command Blocks: Introducing new command blocks with more advanced programming concepts, such as loops, conditionals, and functions, could provide a more challenging and educational experience. This would help players progress from basic to more complex programming skills.

2. Multi-Level Challenges: Creating multi-level challenges with increasing difficulty can help maintain player engagement and provide a clear progression path. Each new level could introduce new obstacles, puzzles, and programming concepts.

3. Interactive Environment Elements: Adding interactive elements to the maze, such as moving platforms, keys and locks, and switches that alter the maze layout, could create more dynamic and interesting puzzles for players to solve.

4. Character Customization: Allowing players to customize the tortoise with different skins, accessories, and animations can make the game more personal and enjoyable. Unlockable customizations could also serve as rewards for completing levels.

Enhanced Educational Value

1. In-Game Tutorials: Implementing interactive tutorials that guide players through the basics of programming and the game mechanics can help new players get started quickly. These tutorials could be designed as part of the initial levels.

2. Progress Tracking and Analytics: Introducing a progress tracking system that monitors players' advancement through the game and provides analytics on their learning can help educators and parents understand how well players are grasping programming concepts. This system could include dashboards and reports.

3. Collaboration and Sharing: Adding features that allow players to share their command sequences and solutions with others could promote collaboration and peer learning. Players could also rate and comment on each other's solutions, fostering a community of learners.

Augmented Reality Enhancements

1. Improved AR Tracking: Enhancing the AR tracking capabilities to make the placement and interaction with the maze more stable and accurate would improve the overall user experience. This could involve using advanced AR frameworks or custom algorithms.

2. Real-World Object Interaction: Allowing the game to recognize and interact with real-world objects placed in the AR environment could add an additional layer of immersion. For example, players could use physical objects to block or guide the tortoise.

VI - Conclusion

The Unity AR Maze Game combines educational content with engaging gameplay, leveraging AR technology to create an immersive learning experience. By using a variety of free Unity assets and custom scripts, the game offers a unique platform for teaching basic programming concepts. Players interact with the 3D maze in their real-world environment, providing a hands-on learning experience that is both fun and educational. This project showcases the potential of AR in educational and entertainment applications, making learning a more interactive and enjoyable process.

At the following link is available: a short video presenting the project, an APK file to install the game on Android devices and the entire solution.