UNIVERSITY OF THESSALY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

DESIGN AND DEVELOPMENT OF A VIRTUAL REALITY EDUCATIONAL APPLICATION WITH 5G NETWORK CAPABILITIES

Master Thesis

Author: Theodosiou Georgios

Supervisor: Korakis Athanasios

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in

Science and Technology of Electrical and Computer Engineering

Department of Electrical and Computer Engineering

Volos 2023

ΠΕΡΙΛΗΨΗ

Αυτή η διατριβή παρουσιάζει την ανάπτυξη και υλοποίηση μιας εφαρμογής εικονικής πραγματικότητας (VR) που σχεδιάστηκε για να ενισχύσει την εκπαίδευση STEM μέσω μιας διαδραστικής εμπειρίας προγραμματισμού. Η εφαρμογή προσφέρει ένα ολοκληρωμένο σύνολο εργαλείων και χαρακτηριστικών που είναι ειδικά προσαρμοσμένα για την εκμάθηση των βασικών αρχών του προγραμματισμού. Η εκπαιδευτική πτυχή της εφαρμογής τονίζεται μέσω της ενσωμάτωσης των αρχών του STEM. Οι χρήστες ενθαρρύνονται να εφαρμόσουν δεξιότητες επίλυσης προβλημάτων, λογικής σκέψης και δημιουργικότητας για να λύσουν διαφορά προβλήματα που αφορούν τον προγραμματισμό. Η εφαρμογή παρέχει μια φιλική προς τον χρήστη διεπαφή και ελευθέρια στον τρόπο χειρισμού της, καθιστώντας την προσιτή σε μαθητές με διαφορετικά επίπεδα ικανοτήτων. Επιπλέον, η εφαρμογή αξιοποιεί την τεχνολογία 5G για να ενισχύσει την εμπειρία μάθησης και προσφέρει την δυνατότητα, πολλαπλοί χρήστες να συνεργάζονται σε πραγματικό χρόνο, παρατηρώντας τις δημιουργίες ο ένας του άλλου και κατά αυτόν τον τρόπο συμμετάσχουν σε κοινές δραστηριότητες προγραμματισμού. Αυτό προάγει την ομαδική εργασία, την ανταλλαγή γνώσεων και την εμβάθυνση της κατανόησης των αρχών του προγραμματισμού. Συνοψίζοντας, η διατριβή αυτή επιδεικνύει τη δυνατότητα της τεχνολογίας της εικονικής πραγματικότητας στην εκπαίδευση STEM, τονίζοντας τα οφέλη της εικονικής πραγματικότητας και τον ρόλο του 5G στη δυνατότητα της συνεργατικής μάθησης.

ABSTRACT

This thesis presents the development and implementation of a virtual reality (VR) application designed to enhance STEM education through an interactive coding experience. The application utilizes the power of VR technology with the principles of coding to provide an engaging and educational platform. The application offers a comprehensive set of tools and features specifically tailored for coding education. The educational aspect of the application is emphasized through the incorporation of STEM principles. Users are encouraged to apply problem-solving skills, logical thinking, and creativity to complete coding challenges and develop their own projects. The application provides a user-friendly interface and intuitive controls, making it accessible to learners of all skill levels. Furthermore, the application leverages 5G technology to enhance the learning experience by enabling multiplayer functionality. Multiple users can collaborate in real-time, observing each other's creations, and engaging in collaborative coding projects. This promotes teamwork, knowledge sharing, and a deeper understanding of coding concepts. Overall, this thesis demonstrates the potential of VR technology in STEM education, highlighting the benefits of VR experiences and the role of 5G in enabling collaborative learning.

Acknowledgements

I would like to express my appreciation to my supervisor Professor Korakis Athanasios, for his support and guidance during the deployment of this thesis. His passion and trust were my motive to do my best and fulfill my thesis with the best possible results. Without his valuable assistance, the completion of this work would be a lot harder.

I am also grateful to my family and friends who have provided me with moral and emotional support in my life. Their spiritual support through every hard time helped me to pass through all the obstacles that appeared along the way.

Table of Contents

List of Figures

List of Abbreviations

3D	Three Dimensional
STEM	Science, Technology, Engineering, and Mathematics
5G	Fifth-generation Network
IOT	Internet Of Things
HMD	Head-Mounted Display
HDMI	High-Definition Multimedia Interface
USB	Universal Serial Bus
PBR	Physically-Based Rendering
UI	User Interface
API	Application Programming Interface
CPU	Central Processing Unit
GPU	Graphics Processing Unit
PUN	Photon Unity Networking
RAM	Random Access Memory
VR	Virtual Reality

1 Introduction and Motivation

1.1 Introduction

Education has always strived to ignite curiosity and create immersive learning experiences. Now, imagine a world where students can step beyond the boundaries of their classrooms and embark on captivating educational journeys. This possibility becomes a reality through the synergy of Virtual Reality (VR) and the transformative power of 5G networks. In this thesis, we explore the design and development of an innovative VR educational application that harnesses the potential of 5G, opening new avenues for engaging and enriching learning environments.

Virtual Reality (VR) is a technology that transports users into virtual worlds, replicating real or imagined environments. With the rapid advancements in VR hardware and software, this transformative technology has gained significant attention across industries, including education. VR offers unprecedented opportunities to create interactive and engaging learning experiences, enabling learners to explore virtual environments, manipulate objects, and collaborate with others. Its growing significance lies in its potential to enhance knowledge retention, foster critical thinking skills, and bridge the gap between theory and practice. As VR continues to evolve, its impact on education holds promise for revolutionizing traditional teaching methods and unlocking new frontiers of learning.

The incorporation of Virtual Reality (VR) in learning brings forth a multitude of advantages over traditional teaching methods. Firstly, VR offers an unparalleled level of immersion and interactivity, enabling students to transcend the confines of textbooks and lectures. Instead, they are transported into virtual worlds where they can engage with realistic simulations and hands-on experiences. This immersive nature of VR fosters active participation and deepens students' understanding by appealing to multiple senses.

Secondly, VR promotes experiential learning, allowing students to apply theoretical knowledge in practical scenarios. They can experiment, make mistakes, and learn from the consequences within the safe confines of a virtual environment. This practical application of concepts enhances critical thinking, problem-solving, and decision-making skills.

Furthermore, VR provides a highly customizable and adaptive learning experience. Students can navigate at their own pace, revisit complex concepts, and receive immediate feedback, fostering personalized learning journeys. This adaptability accommodates various learning styles and individual preferences, promoting a more inclusive educational environment.

Moreover, VR facilitates the exploration of abstract or inaccessible subjects. It can visualize complex processes, microscopic structures, or astronomical phenomena that are otherwise challenging to comprehend. By enabling students to visualize and manipulate these concepts in a virtual space, VR makes abstract subjects more tangible and engaging.

Additionally, VR supports collaborative learning. Students can connect and interact with peers in shared virtual spaces, promoting teamwork, communication, and the exchange of ideas. This collaborative aspect of VR enhances social skills and prepares students for real-world collaboration.

Lastly, VR transcends physical boundaries, granting access to experiences and locations otherwise unavailable due to geographical, financial, or logistical constraints. Students can virtually visit historical landmarks, explore remote ecosystems, or interact with experts from around the world. This global connectivity broadens horizons and fosters cultural understanding.

In summary, the advantages of using VR in learning encompass immersive experiences, experiential learning, adaptability, visualization of abstract concepts, collaborative opportunities, and overcoming physical limitations. By harnessing the power of VR, education can be transformed into a captivating, engaging, and effective process that prepares students for the complexities of the modern world.

In this thesis, I focus on the design and development of a VR educational application that introduces students to the world of coding through an innovative approach. My application leverages the immersive power of VR to create a dynamic learning environment where users can interact with a grid-based landscape and manipulate objects using code. Through an easy-to-use designed interface, users will have access to a set of code tiles, including instructions that specify several attributes of those code tiles, enabling them to craft precise sequences of code. By executing their code, they will witness the instantiation of objects within the virtual landscape, precisely arranged as specified in their written instructions. This implementation aims to bridge the gap between coding theory and practical application, nurturing problem-solving skills and computational thinking in an engaging and visually stimulating VR setting.

1.2 Motivation

The motivation behind developing a Virtual Reality educational application stems from the recognition of the transformative potential that VR holds in revolutionizing the learning experience. Traditional educational approaches often struggle to captivate and engage students, leading to limited interest and retention of information. By leveraging the immersive and interactive nature of VR, an educational application can provide a dynamic and engaging platform for students to explore and learn. By combining the power of VR technology with educational goals, the motivation behind this development is to enhance the educational landscape, inspire curiosity, and empower students to actively participate in their own learning process.

The application of thesis aligns closely with the principles of STEM education, providing an innovative approach to learning. STEM education emphasizes the integration of science, technology, engineering, and mathematics to foster critical thinking, problem-solving, and analytical skills. By utilizing a VR environment where students can manipulate objects and code using tiles, the application facilitates hands-on exploration and experimentation. Through this interactive coding experience, students engage in computational thinking, logical reasoning, and algorithmic problem-solving, which are fundamental aspects of STEM disciplines. The VR tile coding approach encourages students to think systematically, break down complex tasks into smaller steps, and design precise sequences of code to achieve desired outcomes. This promotes computational and mathematical thinking while enhancing logical reasoning and troubleshooting skills. Moreover, the application provides a platform for students to explore engineering concepts by designing and executing code that instantiates objects within the virtual landscape. The integration of VR technology in the context of coding not only makes the learning process more engaging but also cultivates creativity, collaboration, and innovation, all of which are essential components of STEM education.

2 Background

2.1 Introduction to Virtual Reality (VR)

Virtual Reality (VR) is an advanced technology that creates immersive and interactive computer-generated environments, simulating real or imaginary worlds. It enables users to engage with these environments through specialized hardware, such as VR headsets, motion controllers, and haptic feedback devices. The core principle of VR is to provide users with a sense of presence, where they feel as if they have been transported into a different reality. This is achieved by combining stereoscopic visuals, spatial audio, and responsive interactions to create a convincing and engaging experience. By enveloping users in a simulated environment, VR has the power to transcend the limitations of traditional media, offering new possibilities for entertainment, education, training, and various other domains.

The immersive nature of VR experiences holds immense potential in revolutionizing education. Unlike traditional learning methods that rely on passive consumption of information, VR provides a highly interactive and engaging learning environment. By transporting students into virtual worlds, VR offers a unique opportunity for experiential learning, enabling them to actively participate, explore, and manipulate virtual objects and scenarios. This immersive engagement stimulates multiple senses, enhancing knowledge retention and deepening understanding. Students can walk through historical landmarks, dive into the depths of the ocean, or explore distant planets, all from the confines of their classrooms. This level of immersion fosters a sense of presence and emotional connection, creating memorable and impactful learning experiences. VR also enables educators to design simulations and scenarios that replicate real-world challenges, providing students with hands-on practice and critical thinking opportunities. Whether it's conducting virtual science experiments, practicing complex procedures in a safe environment, or collaborating with peers in virtual spaces, VR empowers learners to actively construct knowledge, develop problem-solving skills, and bridge the gap between theory and practice.

Virtual Reality has emerged as a groundbreaking technology with numerous applications across various industries. Its immersive and interactive nature has paved the way for transformative experiences and advancements in fields such as:

1) Healthcare: VR is being used for medical training simulations, surgical planning, and pain management therapies. It provides a safe and controlled environment for medical professionals to practice procedures and enhances patient rehabilitation and mental well-being. As an example, a company named Osso VR uses virtual reality technology to help surgeons practice surgical procedures and interact with medical devices. (ossovr, n.d.)
2) Education: VR is revolutionizing the way students learn by creating engaging and interactive virtual environments. It enables immersive experiences that enhance understanding and retention of complex concepts, especially in subjects like science, history, and geography. For example, the company Tech Row offer students virtual field trips so they can explore historical sites like the Colosseum and Ancient Rome from their classrooms and even embark on educational adventures inside the human body. (techrow, n.d.)

3) Architecture and Design: VR allows architects and designers to create virtual walkthroughs of buildings and spaces before they are constructed. This enables clients and stakeholders to visualize and provide feedback, leading to more efficient design iterations and reducing costs.

2.2 Significance of STEM Education

STEM education, which stands for Science, Technology, Engineering, and Mathematics, plays a vital role in equipping students with the knowledge and skills needed to thrive in an increasingly technological and data-driven world. STEM education goes beyond teaching these disciplines as separate entities; it emphasizes an integrated and interdisciplinary approach to problem-solving and critical thinking. By fostering curiosity, creativity, and analytical thinking, STEM education prepares students to tackle complex challenges and adapt to a rapidly evolving society. Through hands-on experiments, practical applications, and collaborative projects, students gain practical insights into real-world problem-solving and gain a deep understanding of the scientific method, technological innovations, engineering design principles, and mathematical reasoning. STEM education cultivates a strong foundation in these disciplines and nurtures the ability to think critically, analyze data, and propose innovative solutions. As technology continues to shape our world, STEM education has become increasingly important for developing a skilled workforce, promoting innovation, and driving economic growth.

The integration of STEM education with Virtual Reality (VR) technology offers a powerful combination that enhances the learning experience and promotes deeper engagement with STEM subjects. VR provides a unique platform for students to explore complex scientific concepts, experiment with engineering designs, delve into mathematical simulations, and interact with technological innovations in a virtual environment. Through immersive experiences and interactive simulations, students can visualize abstract theories, manipulate virtual objects, and observe cause-and-effect relationships firsthand. This integration bridges the gap between theoretical concepts and practical application, fostering a deeper understanding of STEM subjects and cultivating critical thinking, problem-solving, and analytical skills. VR technology allows students to embark on virtual field trips, conduct virtual experiments, and collaborate with peers in virtual environments, providing an enriched and dynamic learning environment. By combining STEM and VR, educators can create engaging, hands-on learning experiences that inspire curiosity, promote discovery, and empower students to become active participants in their own learning journey.

2.3 Introduction to 5G Network Technology

The fifth-generation (5G) network technology is the next evolutionary step in wireless communication, designed to revolutionize connectivity and enable a wide range of innovative applications. 5G is defined by its high data rates, low latency, and massive device connectivity, offering significant improvements over previous generations. With 5G, users can experience blazing-fast data speeds, enabling rapid downloads, seamless streaming of high-quality multimedia content, and real-time interaction with cloud-based applications. One of the key features of 5G is its low latency, which refers to the minimal delay in data transmission. This ultra-low latency paves the way for highly responsive applications, such as real-time multiplayer gaming, remote surgery, and autonomous vehicles, where instant communication and quick response times are critical. Additionally, 5G supports massive device connectivity, allowing a large number of devices to be connected simultaneously without compromising network performance. This feature is crucial for the Internet of Things (IoT) ecosystem, enabling seamless communication between billions of interconnected devices, from smart homes and wearables to industrial sensors and autonomous systems. Overall, 5G technology promises to unlock new possibilities, enhance user experiences, and enable transformative applications in various domains, including healthcare, transportation, entertainment, and education.

The deployment of 5G networks holds significant potential in enhancing Virtual Reality (VR) experiences by addressing key challenges such as responsiveness and latency. One of the primary advantages of 5G networks is their ability to provide ultra-low latency, allowing for near-instantaneous communication between devices. In the context of VR, this translates to a more immersive and seamless experience, as users can interact with the virtual environment in real time without noticeable delays. The reduced latency ensures that actions and movements in VR are accurately and quickly reflected, enhancing the sense of presence and increasing the level of immersion. Moreover, 5G networks offer substantially improved data rates, enabling the delivery of high-quality and high-resolution content to VR devices. This enables the seamless streaming of VR content, eliminating buffering and providing a smooth experience. Additionally, the increased capacity and connectivity of 5G networks can support multi-user VR experiences, facilitating collaboration and interaction among users in shared virtual spaces. Overall, the integration of 5G networks with VR technology has the potential to revolutionize the way VR experiences are delivered, providing users with enhanced responsiveness, reduced latency, and a more immersive and engaging virtual environment.

2.4 Overview of VR Technologies and Tools

Virtual Reality (VR) technologies encompass a range of hardware and software components that collectively create immersive virtual experiences. One of the key elements is the VR headset, a head-mounted display that covers the user's eyes and provides a stereoscopic visual experience. Popular VR headset models include the Oculus Rift, HTC Vive, and PlayStation VR, each offering high-resolution displays and advanced tracking capabilities. To enhance user interaction within the virtual environment, motion controllers are employed. These handheld devices track the user's hand movements, allowing them to manipulate and interact with virtual objects. Motion controllers, such as the Oculus Touch and HTC Vive controllers, feature buttons, triggers, and sensors to provide intuitive and realistic interactions. Additionally, VR experiences often incorporate tracking systems, such as external sensors or inside-out tracking cameras, to precisely capture the user's movements and position in physical space, enabling them to navigate and explore the virtual world with freedom and accuracy. These technologies work in harmony to create a seamless VR experience, enabling users to feel fully present in the virtual environment and interact with it in a natural and intuitive manner.

To create virtual reality (VR) experiences, content creators rely on a variety of powerful tools and platforms. One prominent category of VR content creation tools is game engines, such as Unity and Unreal Engine. These engines provide a robust and versatile framework for developing interactive VR applications. Unity, a widely used game engine, offers a user-friendly interface, extensive documentation, and a vast asset store that provides pre-built 3D models, animations, and sound effects. It supports cross-platform development, allowing creators to deploy VR experiences on multiple devices. Unreal Engine, another popular game engine, boasts high-quality graphics, advanced physics simulations, and a visual scripting system that enables non-programmers to create interactive experiences. Both Unity and Unreal Engine support VR development through specialized plugins and toolkits, offering features like VR scene editing, VR-specific interactions, and VR performance optimization. These game engines provide a powerful and accessible platform for content creators to design and develop immersive VR experiences without requiring extensive programming knowledge.

2.5 Summary of Technical Aspects

In this section, we delve into the technical aspects of the application, examining the essential tools, software, and hardware employed during the implementation process.

2.5.1 HTC Vive VR Headset

For the implementation of this thesis, the headset used is the HTC VIVE headset. It is a high-end headset powered by SteamVR which uses two base stations wirelessly synchronized for the tracking. This base station offers 360-degree play area tracking coverage. It comes with two controllers, one for each hand. The specifications of the VR headset and the controllers are the following:

- Headset Specs

  Screen: Dual AMOLED 3.6’’ diagonal

  Resolution: 1080 x 1200 pixels per eye (2160 x 1200 pixels combined)

  Refresh rate: 90 Hz

  Field of view: 110 degrees

  Safety features: Chaperone play area boundaries and front-facing camera

  Sensors: SteamVR Tracking, G-sensor, gyroscope, proximity

  Connections: HDMI, USB 2.0, stereo 3.5 mm headphone jack, Power, Bluetooth

  Input: Integrated microphone

Eye Relief: Interpupillary distance and lens distance adjustment

- Controller specs

  Sensors: SteamVR Tracking

  Input: Multifunction trackpad, Grip buttons, dual-stage trigger, System button, Menu button

  Use per charge: Approx. 6 hours

  Connections: Micro-USB charging port

- Tracked area requirements

  Standing / seated: No min. space requirements

  Room-scale: 6'6" x 4'11" min. room size, 11'5" x 11'5" max

Each controller has two buttons, one trackpad, one trigger, and two grip buttons as shown in the figure below.

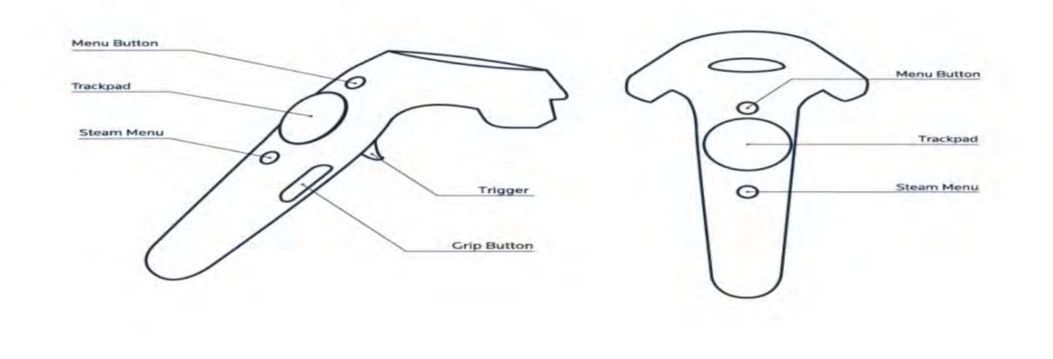


Figure 2-1 VIVE Controller

1. Application Menu Button: Displays the tabbed menu in VR.
2. Touchpad: Usually used for moving around but this depends on the application. It can also be pressed.
3. System Menu Button: display the Steam system menu within VR.
4. Grip Buttons: Their actions depend on the application.
5. Trigger: Action depends on the application.

2.5.2 Unity Game Engine

For the implementation of this thesis, the Unity Game Engine was chosen as the primary tool for developing the VR application. Unity offers a range of key features that are particularly relevant to this project.

One of the notable strengths of Unity is its powerful renderer, which plays a crucial role in creating VR experiences. With its support for physically-based rendering (PBR), dynamic lighting, and post-processing effects, Unity enables developers to deliver high-quality visuals that enhance the realism and immersion of the VR environment.

In terms of scripting, Unity supports multiple languages, including C#, which was utilized in this implementation. This flexibility allowed for the implementation of intricate gameplay mechanics, interactions, and other VR-specific functionalities. Unity's scripting API provides a wide range of functions and classes that facilitated the development of the VR application, enabling seamless integration of VR interactions and logic.

The user interface (UI) of Unity is designed to be user-friendly and intuitive, providing a visual editor that simplifies the development process. This UI allowed for easy placement and manipulation of objects within the VR environment. The drag-and-drop functionality and customizable layout options offered by Unity's UI accelerated the development workflow and enhanced productivity.

Additionally, Unity provides dedicated tools and functionalities for VR development. It offers built-in support for VR platforms and integration with VR headsets, controllers, and tracking systems. This allowed for the integration of VR hardware and the implementation of VR-specific interactions, such as teleportation mechanics and spatial audio.

In conclusion, the Unity Game Engine, with its advanced renderer, versatile scripting methods, user-friendly interface, and dedicated VR development tools, was a suitable choice for implementing this VR application.

2.5.3 Photon Cloud

For the networking aspect of this implementation, the Photon Engine was utilized as a key technology to enable multiplayer functionality and real-time communication within the VR application. The Photon Engine offers a robust and scalable networking solution specifically designed for game development.

One of the notable features of the Photon Engine is its reliability and stability. It leverages a client-server architecture, where clients connect to dedicated servers provided by Photon. This architecture ensures that the network communication is handled efficiently, reducing latency and ensuring smooth synchronization between players. Additionally, Photon's servers are optimized for high performance and can handle a large number of concurrent connections, making it suitable for multiplayer VR experiences with multiple participants.

Photon provides an easy-to-use and flexible API that simplifies the implementation of networking functionality. The API allows developers to send and receive data between connected clients, enabling real-time interactions and synchronization. This was crucial for enabling collaborative experiences and ensuring that all participants in the VR application could interact and see each other's actions in real-time.

Another important feature of the Photon Engine is its support for various networking models. It offers both authoritative and peer-to-peer networking options, allowing developers to choose the model that best fits their specific requirements. The authoritative model, where the server has the final say on game state and actions, is often preferred for multiplayer VR applications to maintain fairness and prevent cheating. However, in certain scenarios, peer-to-peer networking can be utilized to reduce server load and provide a more decentralized networking approach.

Photon also offers a range of networking optimizations to ensure efficient data transmission. It utilizes compression techniques and smart data packaging to minimize bandwidth usage and reduce network congestion. This is particularly important in VR applications, where maintaining a smooth and responsive experience is crucial for user immersion and engagement.

Furthermore, Photon provides robust matchmaking and lobby management functionalities. It allows players to join specific rooms or create their own, enabling matchmaking and grouping of players. This was instrumental in facilitating multiplayer experiences within the VR application, as it ensured that players could easily connect with each other and engage in collaborative coding activities.

Additionally, Photon offers cross-platform compatibility, supporting various platforms and devices. This flexibility allowed for the implementation of the VR application across multiple VR headsets and devices, ensuring a wider reach and accessibility for users.

In conclusion, the Photon Engine provided a comprehensive and powerful networking solution for this implementation. Its reliable client-server architecture, easy-to-use API, support for various networking models, networking optimizations, matchmaking capabilities, and cross-platform compatibility were key technical factors that contributed to the successful implementation of multiplayer functionality in the VR application.

2.5.4 PC Requirements

In order to ensure optimal performance and a seamless user experience, it is essential to consider the minimum PC requirements for running the VR application developed in this thesis. The VR technology employed in the application places certain demands on the hardware, and it is important to provide users with clear guidelines regarding the specifications of the PC system needed to run the application smoothly.

To achieve optimal performance and ensure an immersive experience, the following minimum PC requirements are recommended for running the VR application:

GPU: NVIDIA® GeForce® GTX 1060 or AMD Radeon™ RX 480, equivalent or better.
CPU: Intel® Core™ i5-4590 or AMD FX™ 8350, equivalent or better
RAM: 8 GB RAM or more
Video out: HDMI 1.4, DisplayPort 1.2 or newer
USB Ports: 1x USB 2.0 or better port
Operating system: Windows 10

It is important to note that these are the minimum recommended specifications, and higher-end hardware configurations may deliver even better performance and visual fidelity. Users with systems that exceed the minimum requirements may benefit from enhanced graphics, smoother framerates, and improved overall VR experience.

3 Application

3.1 Description

The application developed for this thesis provides an immersive learning experience, allowing students to witness the immediate impact of their code within a virtual landscape. By executing their code, students can observe the creation of objects within the virtual environment, precisely arranged as instructed in their written code. This interactive feedback mechanism reinforces the understanding of coding concepts such as loops, conditionals, and functions. Moreover, the application goes beyond the syntax and semantics of coding by fostering logical reasoning and algorithmic thinking. Through engaging challenges and activities, students are encouraged to think critically and solve problems using coding principles. By combining hands-on experimentation with visual stimulation, the application offers a unique opportunity for students to develop their coding skills in an immersive and enjoyable manner.

3.2 Functionality and Features

The VR application developed for this thesis comprises two distinct scenes, each serving a specific purpose. The first scene is the main menu, which currently functions as a welcoming and transitional scene to the main experience. In its current state, the main menu provides users with an introduction to the application and serves as a gateway to the learning experience. However, as part of future work, additional features will be incorporated into the main menu, such as a settings tab and other interactive elements, to enhance user control and customization.

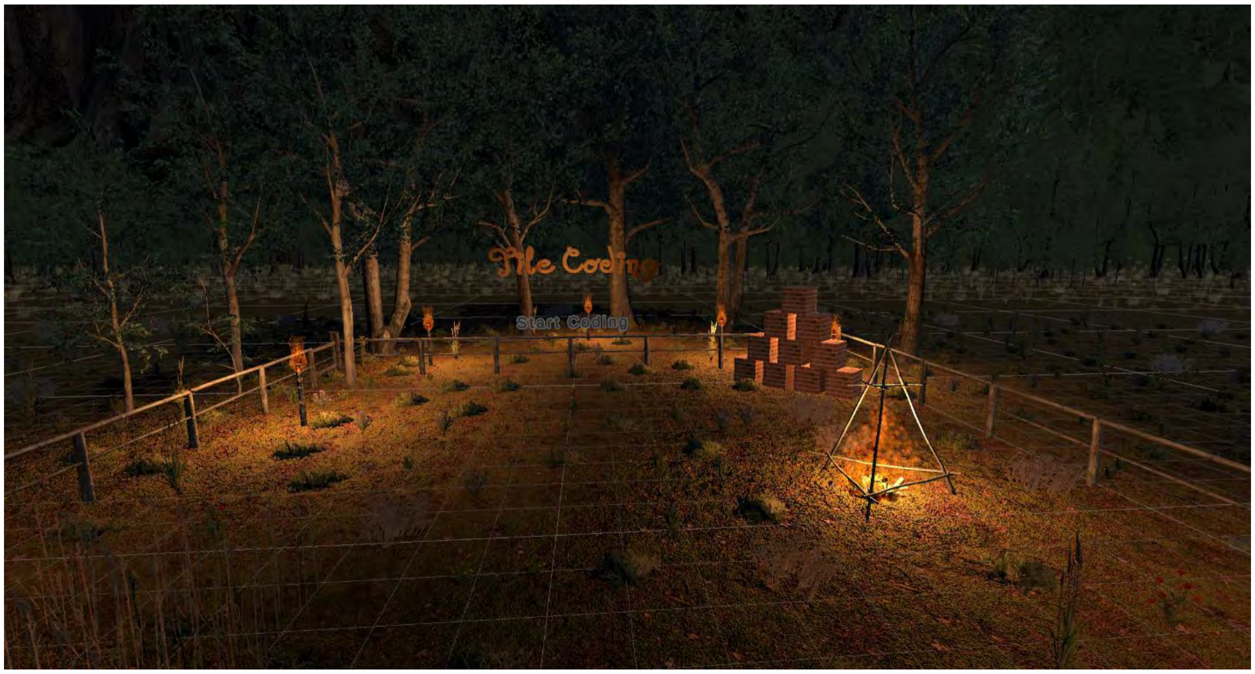
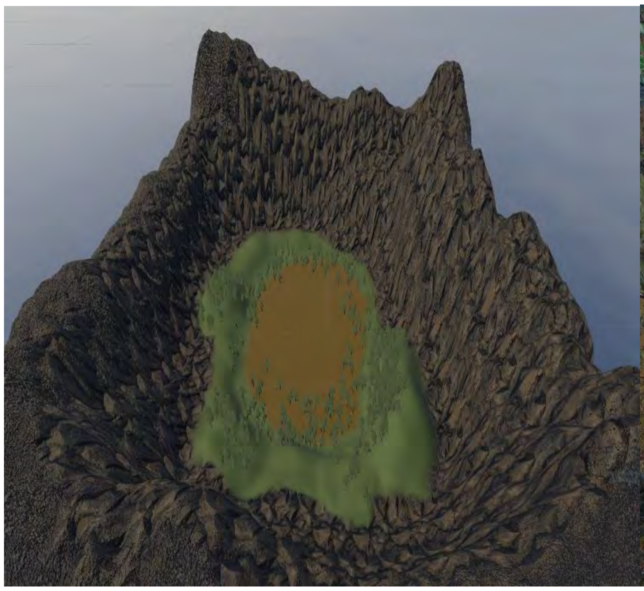


Figure 3-1 Main Menu

The second scene serves as the core of the application, providing students with the primary learning environment. Within this scene, students can actively engage in coding activities and interact with a virtual landscape. This scene offers a range of functionalities and serves as the primary learning environment for students.

The virtual reality (VR) application developed in this thesis takes place in an environment set in an open field surrounded by mountains. Within this virtual field, the player interacts with the environment using hand controllers, specifically the right controller. By pointing the right controller downwards, a ray is emitted from the controller towards the ground. The application registers the point where the ray intersects with the ground, and an 11 by 11 grid appears, visually indicating the chosen interaction area for coding activities.



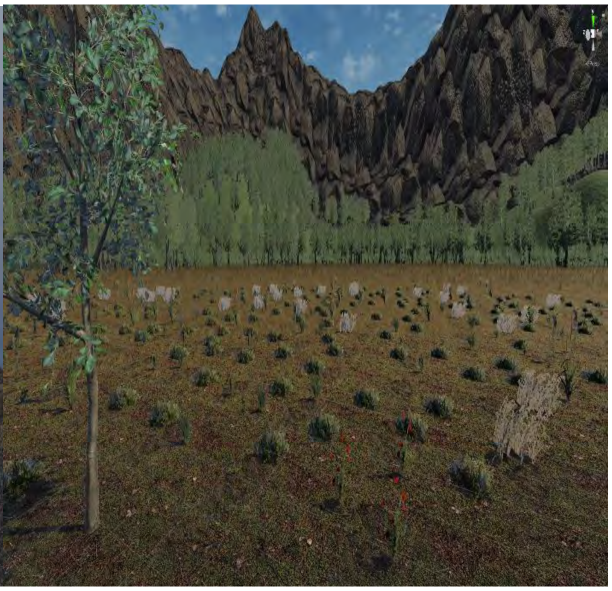


Figure 3-2 Environment Top View

Figure 3-3Environment Player View

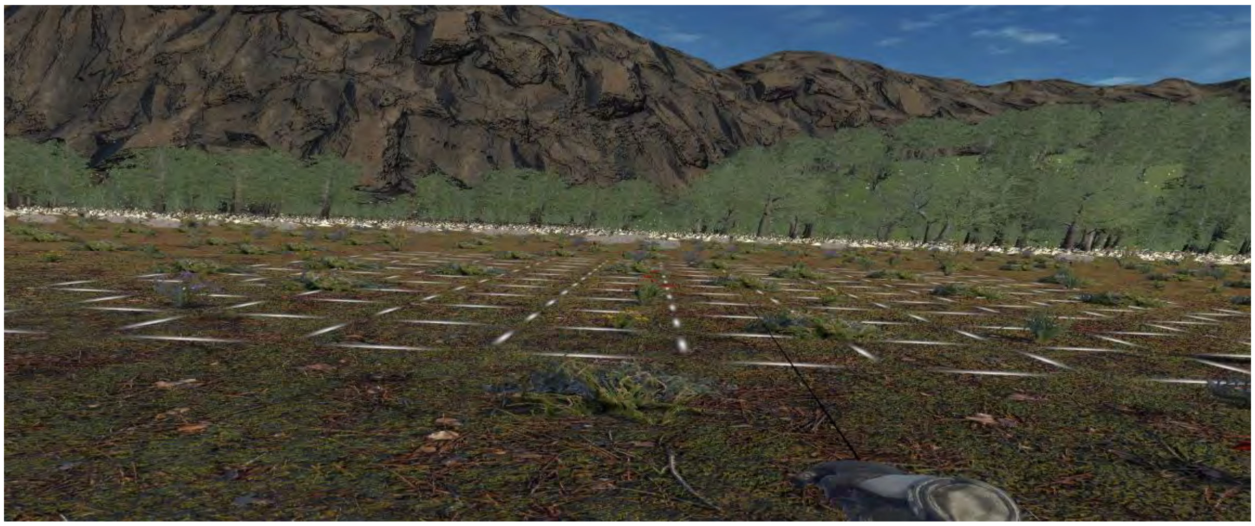


Figure 3-4 Grid

To enhance the player's mobility within the virtual field, the application incorporates a teleportation mechanism. By utilizing the dedicated teleport button, the player can seamlessly transport themselves to different locations within the field, offering flexibility and exploration opportunities.

Once the player has identified their desired interaction area, they have the option to lock the grid in place by pressing the trigger button on the controller. Locking the grid activates the coding interface, which provides the necessary tools for creating code sequences and engaging in coding activities. It is at this point that the player's position within the virtual field becomes fixed, preventing further movement. However, if the player wishes to navigate around the field once again, they can simply press the "Remove grid" button. This action removes the previously selected grid, releasing the player's position and allowing them to freely navigate and explore the virtual environment.

The coding interface of the VR application comprises two panels and an 11x11 grid, systematically numbered from 0 to 120. The right panel, which contains various code blocks represented by blue and yellow blocks. These blocks serve specific functions within the coding environment, with the yellow blocks representing the "Repeat" command for loop execution and the blue blocks representing the terminal "Add" command. In addition to these code blocks, the right panel also provides three directional parameters—horizontal, vertical, and upwards—that are essential for the "Repeat" command. These directional parameters enable the player to specify the direction of a particular loop within their code sequence. The coding interface further incorporates a diverse range of objects that can be utilized with the terminal "Add" command. These objects serve as parameters within the "Add" command, allowing the player to define and instantiate specific objects based on their code instructions. By leveraging this functionality, the player gains the ability to create and manipulate virtual objects within the coding environment, facilitating a hands-on approach to learning coding principles.

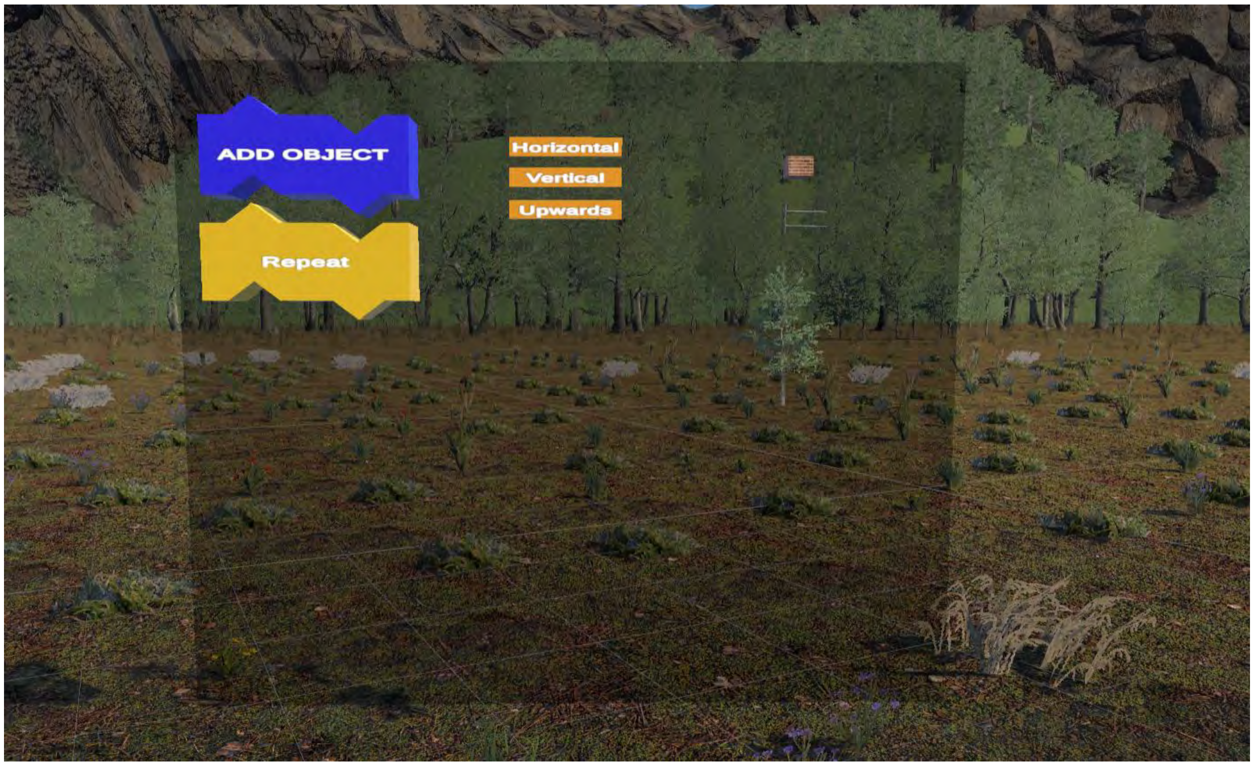


Figure 3-5 Right Panel

The left panel of the coding interface is dedicated to execution and functional controls, along with essential editing capabilities. It hosts a range of buttons that enable various actions within the coding environment. These buttons include the "Compile" button, which validates the code syntax and ensures its readiness for execution. The "Run Code" button initiates the execution of the code sequence, while the "Step" button enables a step-by-step execution, allowing the player to observe the code's behavior at each stage. Additionally, the "Remove Grid" button serves as a navigation tool, unlocking the player's position and removing the previously selected grid for free exploration within the virtual environment. Lastly, the "Reset Code" button enables the player to start afresh by clearing the existing code.

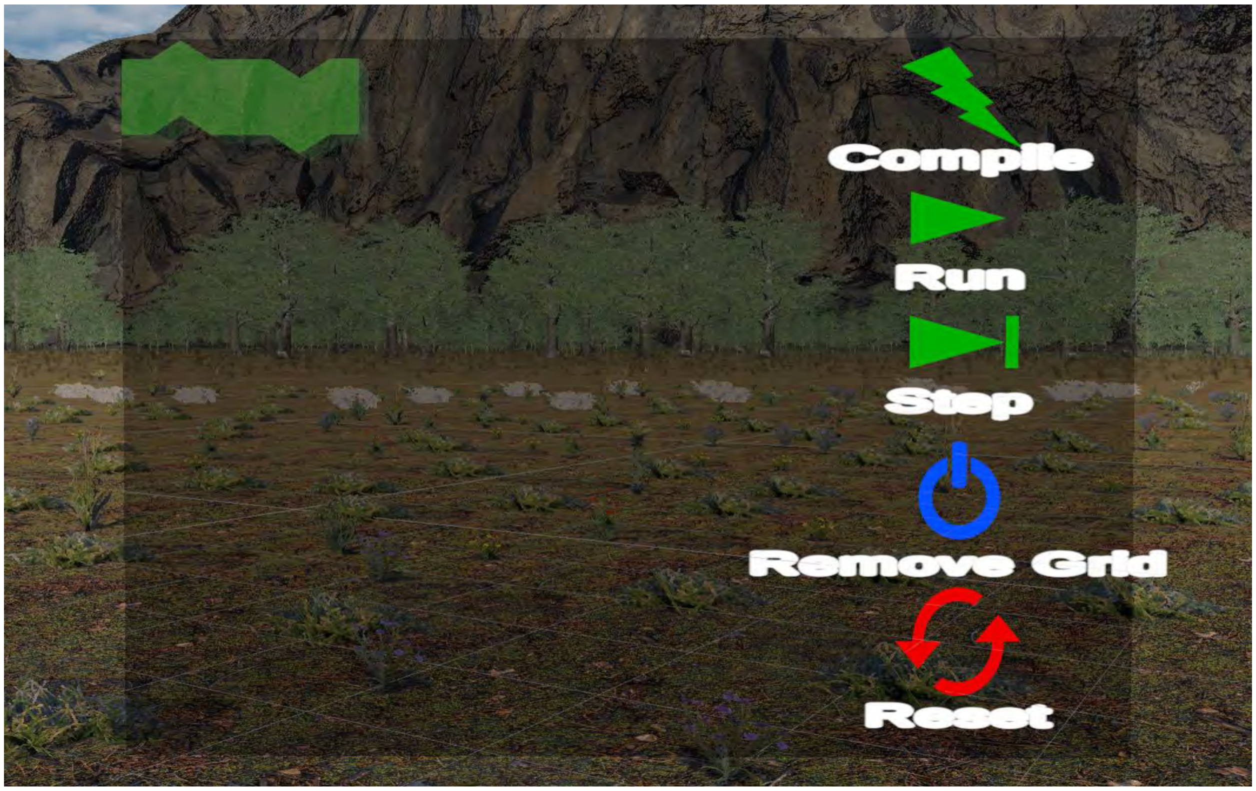


Figure 3-6 Left Panel

In addition to the functional buttons, the left panel also incorporates free slots represented by transparent green blocks. These slots serve as the code editor, allowing the player to drag and drop the code blocks from the right panel into the slots to create their code sequence. This intuitive interaction fosters a hands-on coding experience, empowering students to assemble their code logic by visually arranging the blocks within the code editor.

The player's interaction with the coding interface and the grid is facilitated by a ray that emanates from their right controller. Using this ray, the player can precisely target specific blocks on the interface panels. By directing the ray towards a desired block, the user can select it by pressing and holding the trigger button on the controller, allowing for seamless interaction. To incorporate a code block into the code sequence, the player can grab and drop a desired code block from the right panel (blue and yellow blocks) into a free slot within the left panel (transparent green block), effectively integrating the selected command into the code editor. This intuitive process empowers the player to dynamically assemble their code sequence, fostering a hands-on approach to coding.

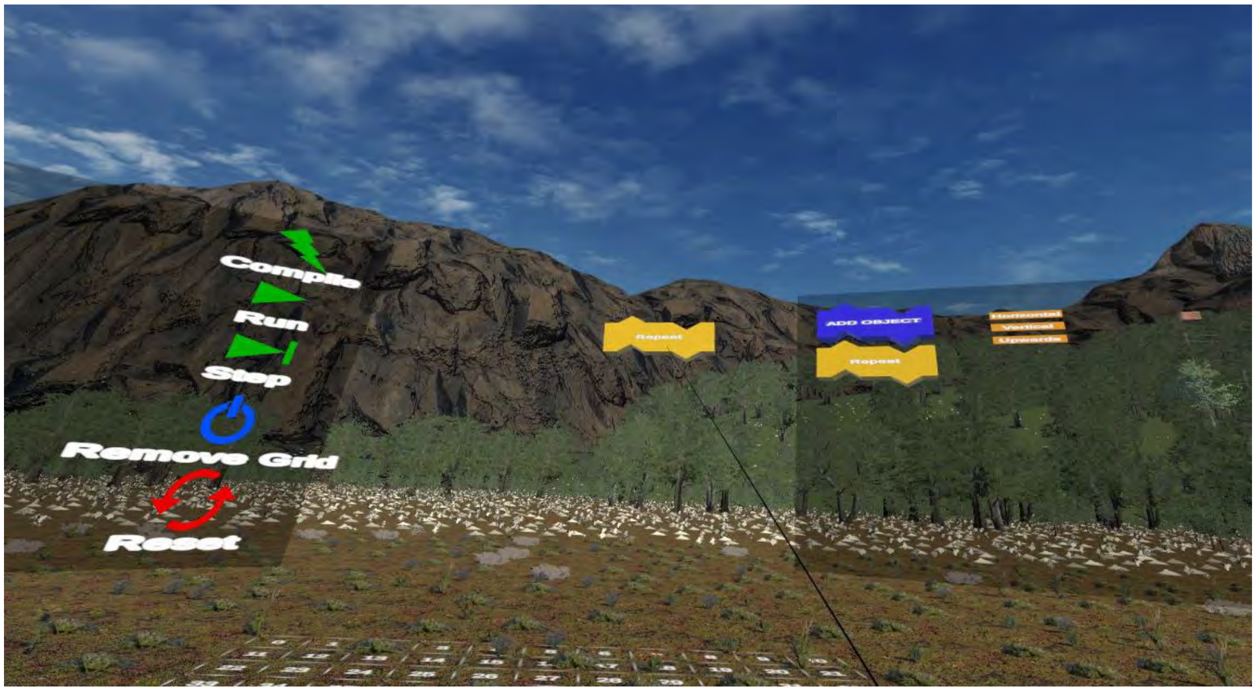


Figure 3-7 Interaction with Interface

Furthermore, the same interaction mechanism applies to setting parameters within the code blocks. The player can use the ray to point at the directional parameters (horizontal, vertical, and upwards) and the objects required as parameters for the code blocks. By selecting and assigning these parameters using the trigger button, the player defines the specific behaviors and properties of the code blocks, enhancing the customization and flexibility of the code sequence.

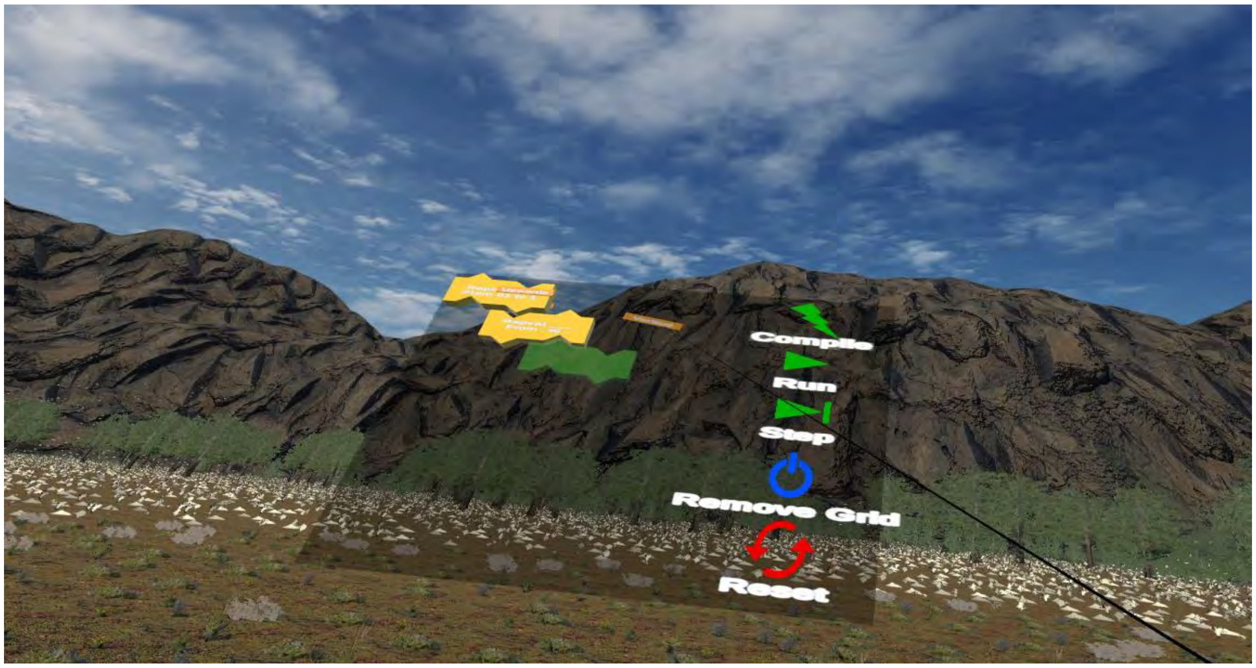


Figure 3-8 Set Parameter

In addition to interacting with the interface panels, the player can also employ the ray to target specific numbers on the grid. By pressing the trigger button, the player can select a particular square on the grid when necessary, such as specifying a range (e.g., 0 to 8) for a loop. This precise selection mechanism ensures accuracy and control in defining the grid-based interactions within the coding environment.

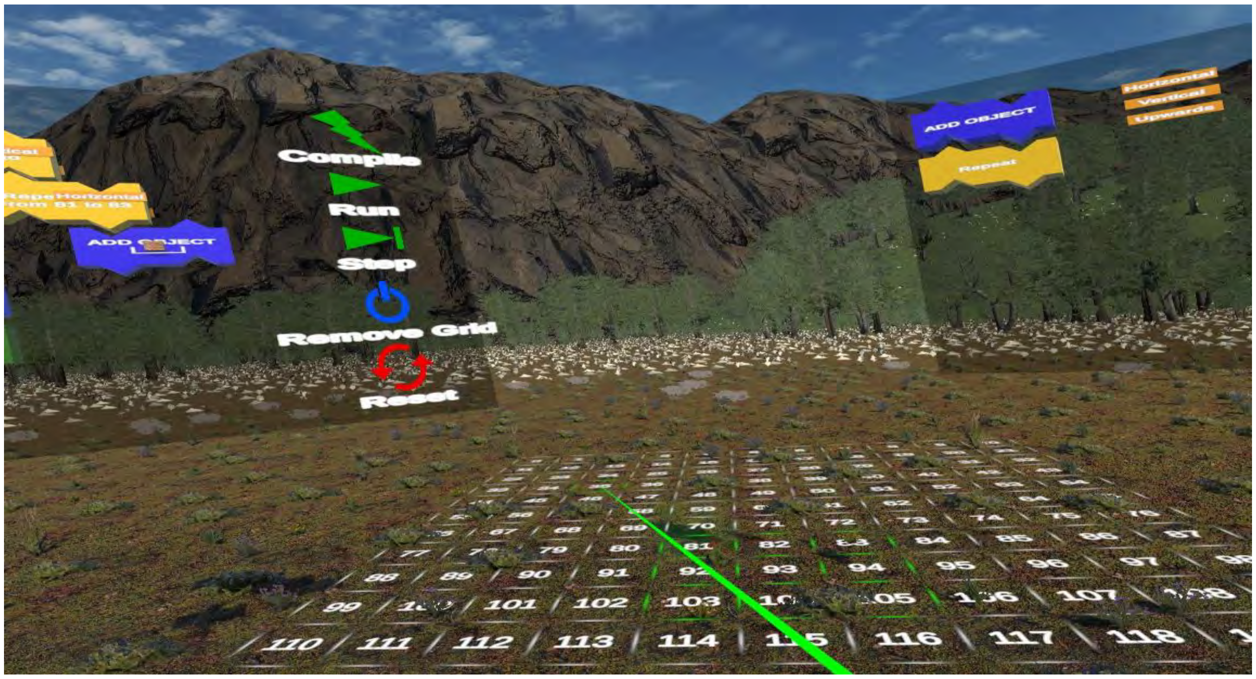


Figure 3-9 Grid Interaction

Creating a code within the VR application follows specific rules to ensure structured and effective coding practices. In our case, the following rules are enforced:

1) Each "Repeat" command requires the user to specify both its direction and range. For instance, the user can specify a horizontal repeat using the syntax "Repeat Horizontal from 90 to 95." In response, the corresponding squares on the grid, ranging from 90 to 95, will be highlighted in green, indicating their selection.
2) Every loop must conclude with a terminal "Add" command. This ensures that the code execution is properly concluded and any necessary actions are performed.
3) In the case of a non-nested "Add" command, the user needs to specify a grid number and an object as parameters. The grid number indicates the position on the grid where the object will be created using the "Add" function.
4) For nested "Add" commands within a loop, only the object to be created needs to be specified. The positional parameter inherited from the loop determines the position on the grid for each iteration.
5) Nesting "Repeat" commands with the same direction parameter is not allowed. This restriction prevents conflicts and ensures proper code structure.

The VR application is designed to enforce these rules, preventing users from proceeding with incorrect or incomplete code. For example, if a user attempts to add a direction parameter to a nested "Repeat" command that duplicates the direction inherited from its parent, the application will reject the redundant parameter and prompt the user to provide a suitable direction. Similarly, the compiler within the application identifies and highlights errors when the user attempts to compile the code, such as incomplete code segments.

Once the user has written their code without any errors, they can proceed by pressing the "Compile" button. Upon doing so, a message will appear confirming that the compilation process is complete, indicating that the code is error-free and ready for execution. The message "Done Compiling" serves as a signal to the user.

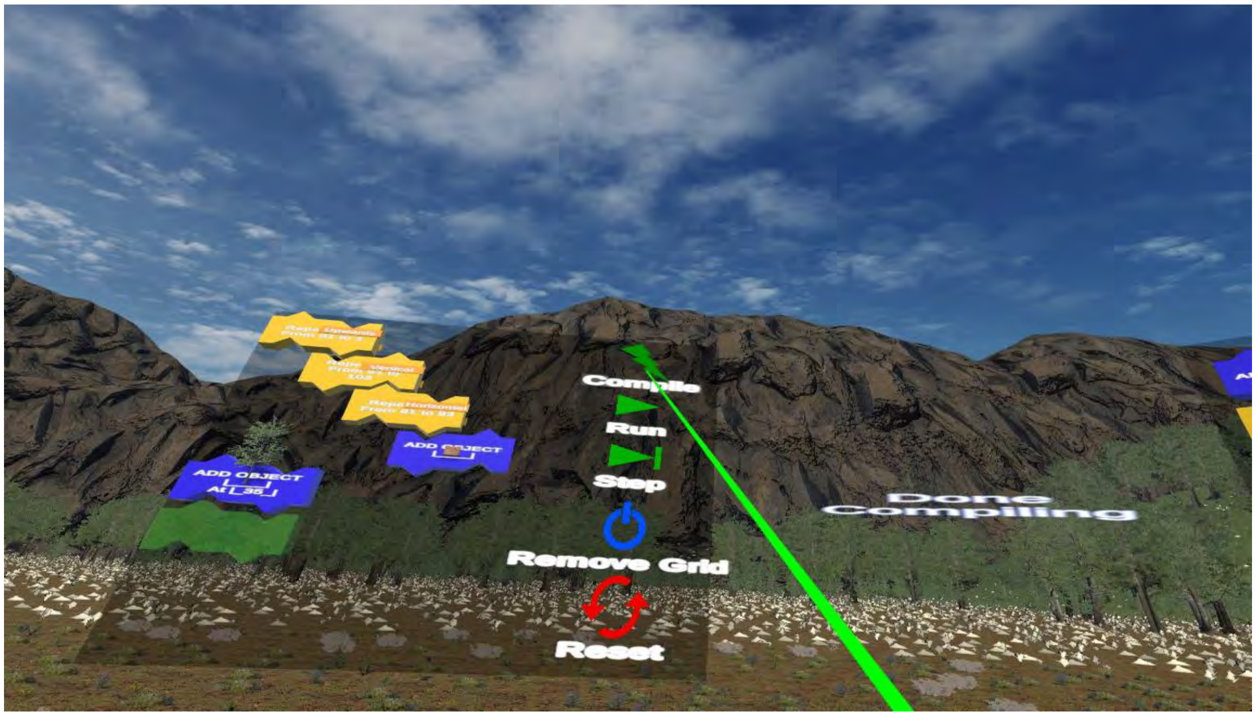


Figure 3-10 Compile

To execute the code, the user can press the "Run" button. The application will then commence the execution process, instantiating each specified object one by one. A small delay of a few milliseconds is introduced between each object instantiation to ensure a clear visual representation of the code's execution. This approach enables the user to observe the order in which each line (block) of code is executed, creating an animated sequence.

Furthermore, for a more detailed analysis of the code's behavior, the user can opt for a step-by-step execution by pressing the "Step" button. With each press of the step button, the application advances to the next line (block) of code, executing it accordingly. This feature provides a valuable learning experience as users gain a deeper understanding of coding principles by visualizing how their code behaves in different scenarios.

To provide flexibility and ease of use, the application offers additional features for code management and navigation. The user has the option to reset their code and start afresh by simply pressing the "Reset" button. This functionality allows them to clear their code and begin coding from a clean slate.

For enhanced exploration of their creation, the user can utilize the "Remove Grid" button. Pressing this button unlocks the player's position and removes the code interface and grid from the scene. Consequently, the user gains the freedom to navigate unrestrictedly within the virtual field, exploring the objects created by their code from various perspectives.

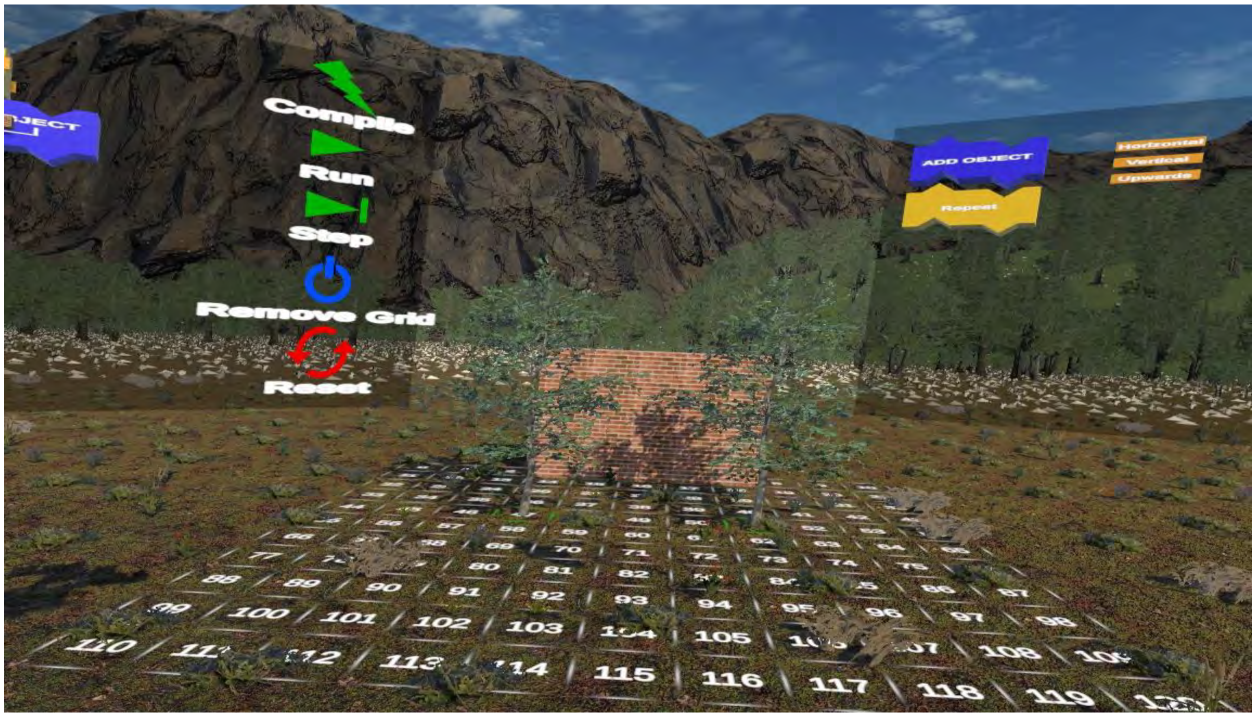


Figure 3-11 Executing the Code

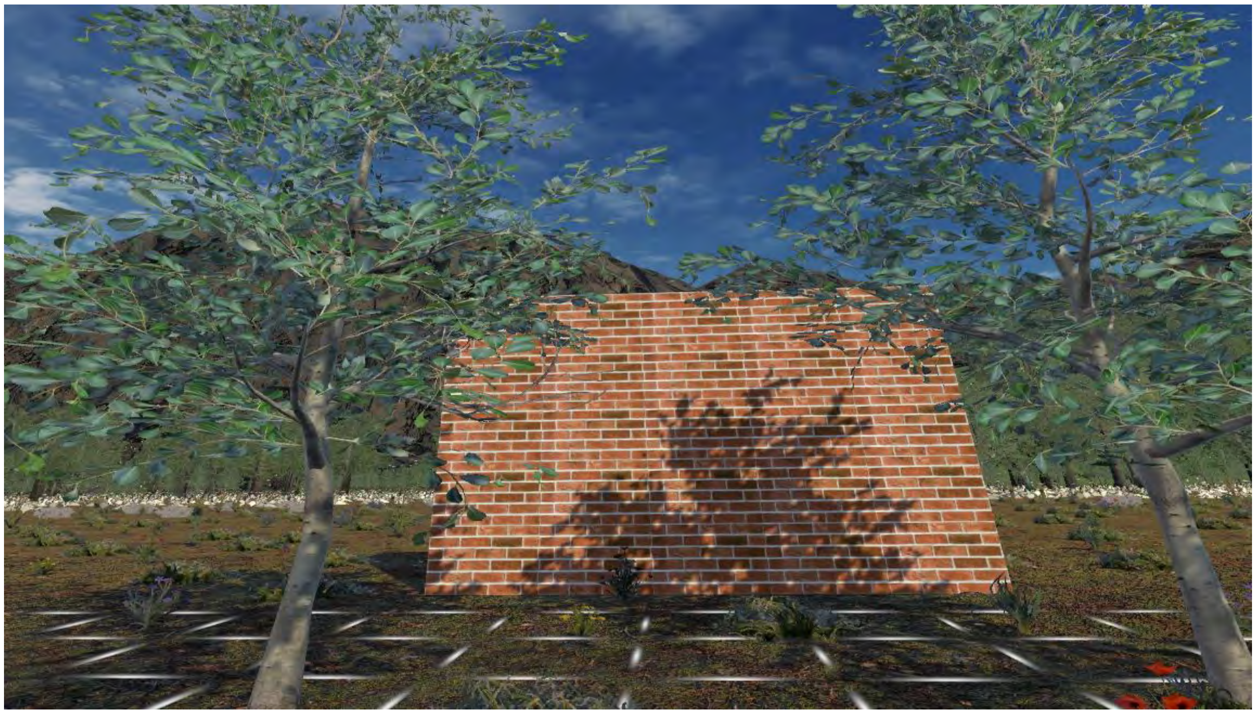


Figure 3-12 Navigation

To harness the potential of 5G technology, the application incorporates a multiplayer feature, enabling simultaneous participation of multiple players within the same virtual experience. This functionality allows each player to engage in independent coding and unleash their creativity by creating their unique creations.

Through the multiplayer aspect, participants have the opportunity to observe and appreciate the creations of other players in real time. This fosters a collaborative environment where users can share ideas, learn from each other's coding techniques, and draw inspiration from the diverse range of creations. The ability to witness the creations of fellow participants adds an interactive and dynamic element to the experience, encouraging engagement and sparking new ideas.

The multiplayer functionality not only promotes a sense of community and collaboration but also provides an avenue for users to showcase their coding skills and creativity. It opens up avenues for peer-to-peer learning, fostering a rich and interactive environment where participants can collectively enhance their understanding of coding principles and expand their repertoire of programming techniques.

3.3 Contribution on Teaching

The VR educational application provides a unique and immersive learning experience for students, offering several benefits in teaching coding principles. By utilizing the interactive and dynamic virtual landscape, students can actively engage in coding activities and experiment with different coding sequences.

One of the key benefits of the application is its ability to visually demonstrate the execution of the code. This visual representation allows students to observe the immediate impact of their code and reinforces the cause-and-effect relationship in coding. By witnessing the objects being created and arranged in the virtual environment, students can better understand the logical flow of their code and develop a deeper comprehension of coding concepts.

The application's step-by-step execution feature further enhances the learning process. By executing the code line by line, students can observe how each block of code contributes to the overall outcome. This feature promotes a detailed understanding of coding principles and encourages students to analyze and troubleshoot their code effectively.

Furthermore, the ability to navigate freely within the virtual landscape after code execution allows students to explore and interact with the objects they have created. This experiential learning approach encourages students to take ownership of their code and fosters creativity and critical thinking.

By leveraging the power of 5G technology, the application's multiplayer functionality offers a unique and enriching experience that transcends individual coding endeavors, fostering a vibrant community of creators who can learn, share, and inspire one another.

Overall, the VR educational application offers an engaging and visually stimulating learning environment that promotes the understanding of coding concepts, logical reasoning, and algorithmic thinking. By combining interactive coding activities, visual feedback, and hands-on exploration, the application empowers students to develop essential coding skills and cultivates a passion for STEM disciplines.

4 Implementation

The implementation of the VR educational application primarily relied on the Unity Game Engine, which served as the main tool for development. The implementation process can be divided into four key parts, each contributing to the overall functionality and experience of the application.

1) The VR aspect
2) The 3D environment aspect
3) The development of the main functionality of the VR application
4) The Networking aspect

Now, let's delve into each section in greater detail.

4.1 VR Aspect

As mentioned previously, the VR aspect of the application plays a pivotal role in its functionality. To bring this aspect to life, I utilized the "OpenVR" software development kit (SDK) and the "Steam VR" Unity plug-in. OpenVR is a versatile SDK and application programming interface (API) developed by Valve, specifically designed to support multiple virtual reality headsets (VR) devices. The SteamVR platform, in turn, utilizes OpenVR as its default API and runtime.

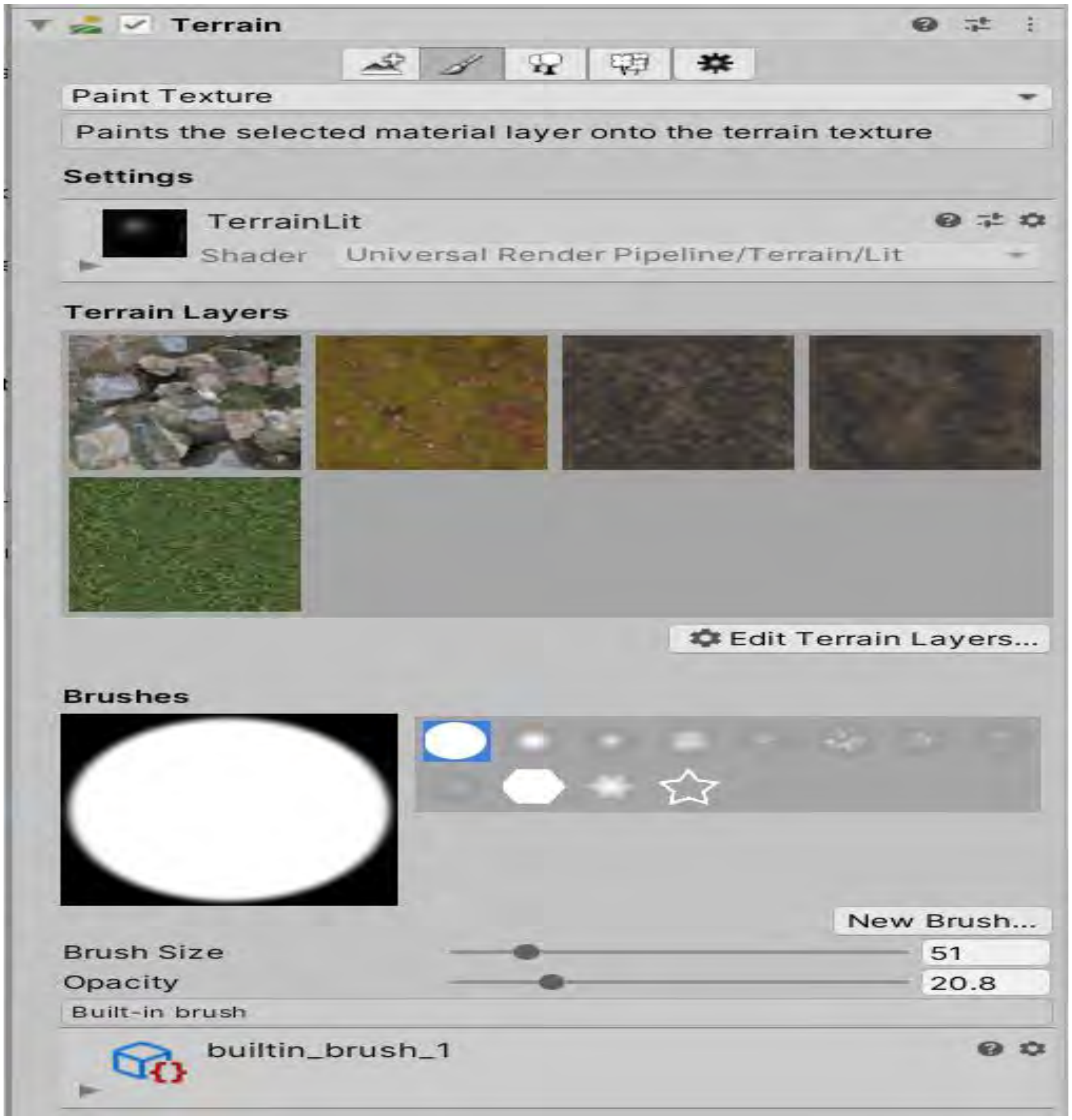
By leveraging these tools, my software was able to establish communication with the VR hardware, facilitating the handling of user movements and inputs from their VR headset and controllers. This integration enabled an interactive experience for the users, as they could navigate and interact with the virtual environment in a natural and intuitive manner.

To ensure the scene was rendered effectively within the VR headset, I employed Unity's Multi-pass stereo rendering mode. This mode employs a dedicated render pass for each eye of the headset, encompassing every head-mounted display. By leveraging this approach, I was able to optimize the rendering process by sharing certain components of the render loop, such as scene culling and shadow maps, between both eyes. This not only improved efficiency but also enhanced the overall visual quality, providing users with a compelling virtual environment. (unity3d, n.d.)

4.2 3D Environment

To develop the virtual environment for the application, I leveraged the built-in tools provided by the Unity game engine. Specifically, for the terrain creation, I utilized Unity's terrain tool, which offers a range of functionalities to shape the environment according to specific requirements. The terrain refers to the virtual landscape or surface on which the virtual environment is built, providing a foundation for the placement of objects, textures, and other environmental elements. With this tool, I was able to sculpt and mold the terrain to achieve the desired landscape. (unity3d, n.d.)

To add textures to the terrain, I employed the "paint texture" tool provided by Unity. This tool allowed me to select and apply textures, such as grass, by using a brush-like tool to paint the desired texture onto the surface of the terrain. This enabled me to create visually appealing and realistic landscapes with varied textures.



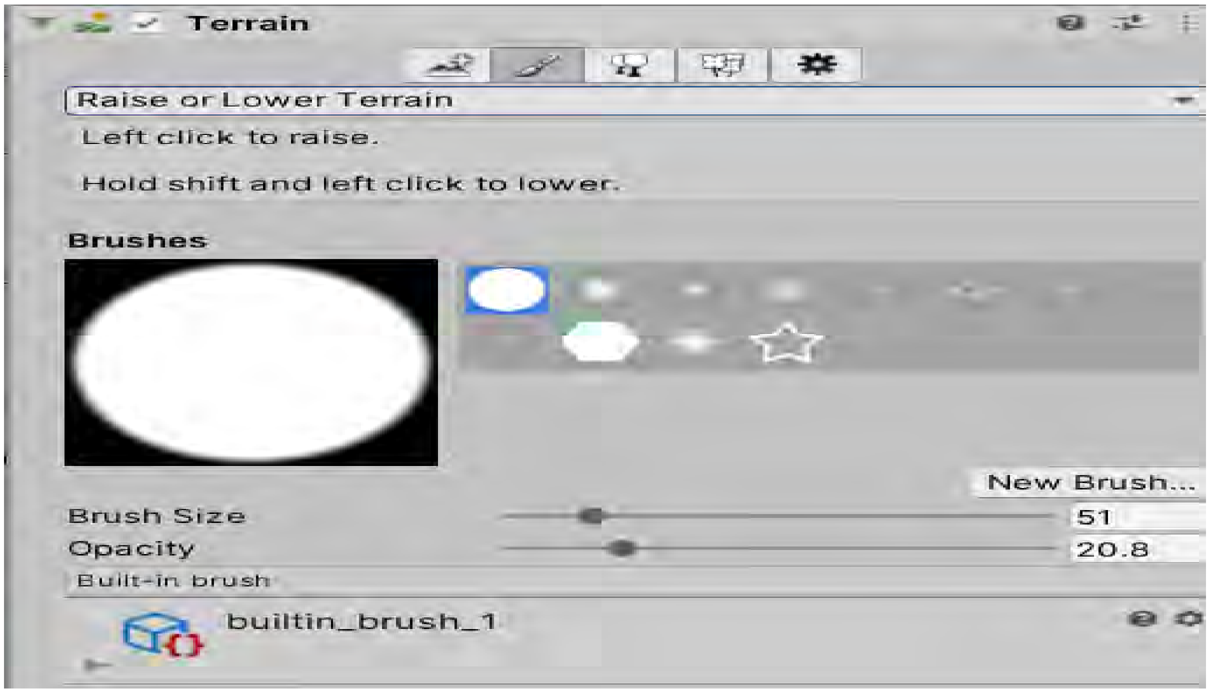
Paints the selected material layer onto the terrain texture

TerrainLit
Shader Universal Render Pipeline/Terrain/Lit

Brush Size 51
Opacity 20.8
Built-in brush
builtin_brush_1

Figure 4-1 Paint Texture Tool

For the creation of the terrain's features, such as mountains and fields, I utilized the "Raise or Lower Terrain" tool. With this tool, I could manipulate the elevation of the terrain using a brush, effectively shaping the heightmap and achieving the desired topography.


Terrain

Raise or Lower Terrain

Left click to raise.

Hold shift and left click to lower.

Brushes

New Brush...

Brush Size 51

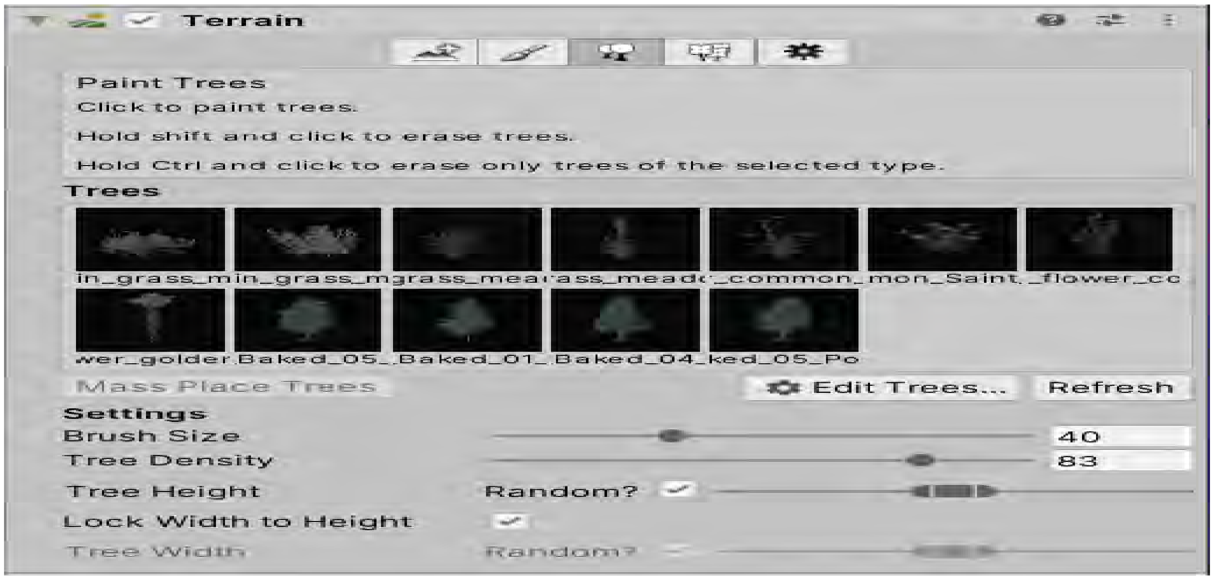
Opacity 20.8

Built-in brush

builtin_brush_1

Figure 4-2 Raise and Lower Terrain Tool

To populate the environment with trees and grass, I employed the "paint trees" function available in Unity. This functionality enabled me to place trees and grass assets throughout the terrain, creating a lush and vibrant virtual environment.


Terrain

Paint Trees

Click to paint trees.

Hold shift and click to erase trees.

Hold Ctrl and click to erase only trees of the selected type.

Trees

in_grass_m in_grass_m grass_mea ass_meadc _common_ mon_Saint _flower_cc

wer_golder Baked_05_ Baked_01_ Baked_04_ ked_05_Po

Mass Place Trees

Edit Trees... Refresh

Settings

Brush Size 40

Tree Density 83

Tree Height Random? ✓

Lock Width to Height ✓

Tree Width Random?

Figure 4-3 Paint Trees Tool

In addition to the terrain, I utilized Unity's "ProBuilder" tool for the creation of various 3D objects, such as the code blocks. ProBuilder is an integrated tool within Unity's editor that allows for the creation and customization of 3D objects. By leveraging extrusion and inset techniques, I could generate a diverse range of game objects suitable for use within the application.

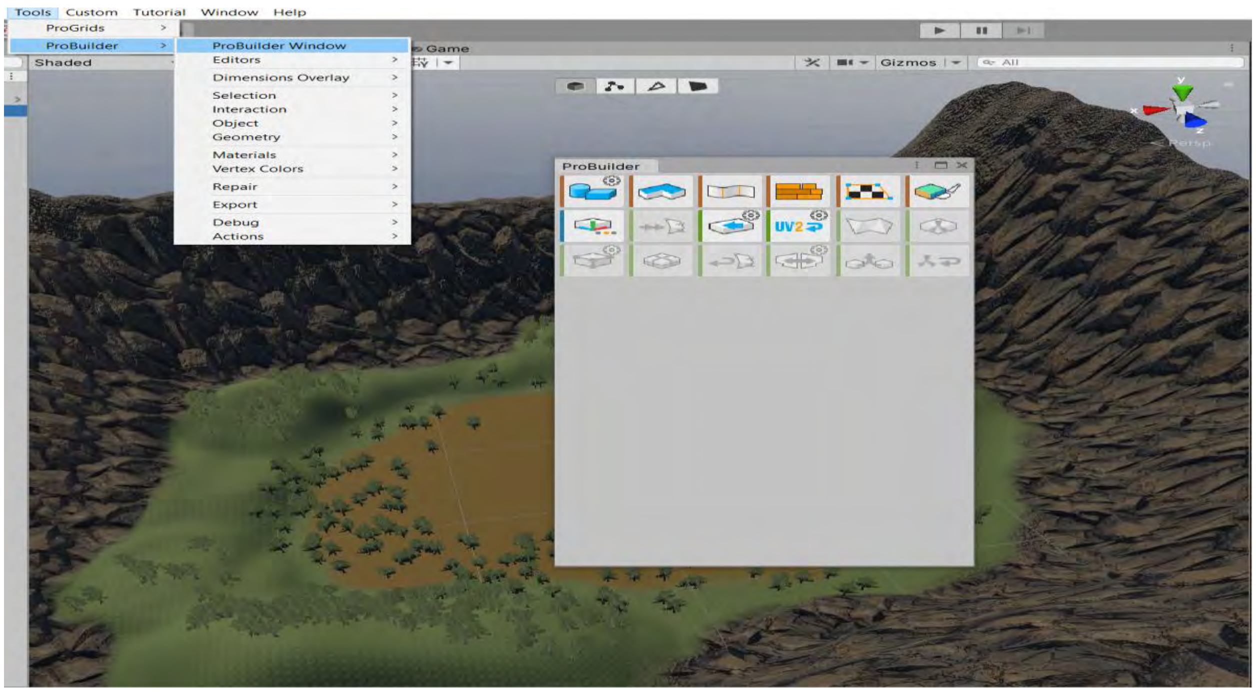
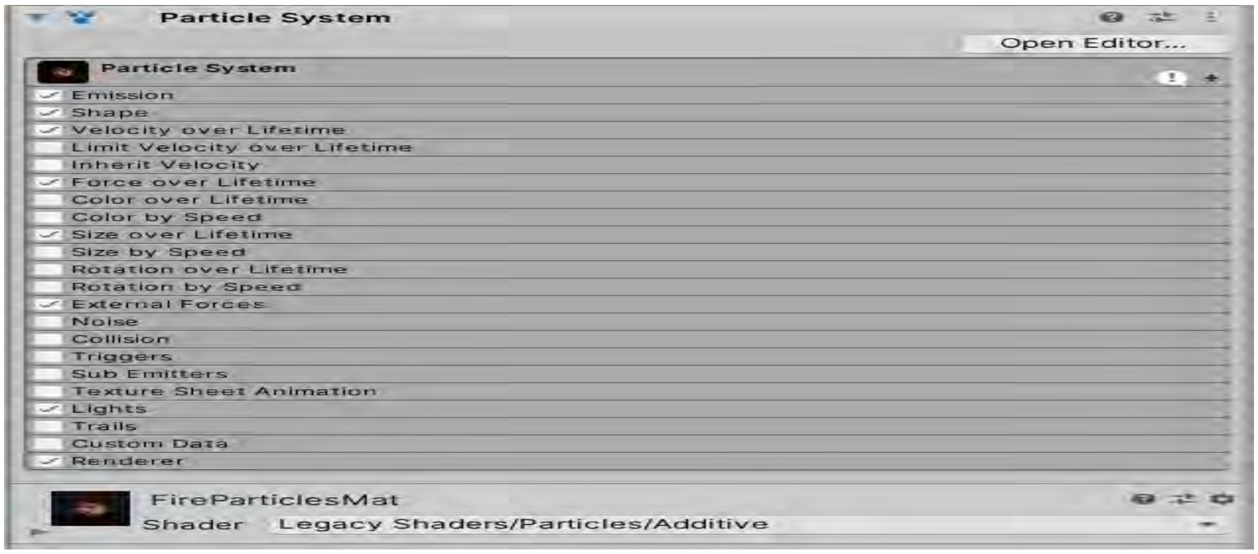


Figure 4-4 Pro Builder

To enhance the visual effects (VFX) in various scenes, such as the fire effect or smoke effect in the main menu, I utilized Unity's Particle System. The Particle System in Unity is a powerful tool that allows developers to generate and control various particle effects, such as smoke, fire, sparks, and more. It simulates the behavior and appearance of particles, providing realistic and dynamic visual effects within the virtual environment. By adjusting parameters like particle shape, size, color, velocity, and lifetime, developers can create captivating and immersive VFX that enhance the overall visual quality of the application.



Particle System

Open Editor...

Particle System
- ✓ Emission
- ✓ Shape
- ✓ Velocity over Lifetime
- Limit Velocity over Lifetime
- Inherit Velocity
- ✓ Force over Lifetime
- Color over Lifetime
- Color by Speed
- ✓ Size over Lifetime
- Size by Speed
- Rotation over Lifetime
- Rotation by Speed
- ✓ External Forces
- Noise
- Collision
- Triggers
- Sub Emitters
- Texture Sheet Animation
- ✓ Lights
- Trails
- Custom Data
- ✓ Renderer

FireParticlesMat

Shader Legacy Shaders/Particles/Additive

Figure 4-5 Unity's Particle System

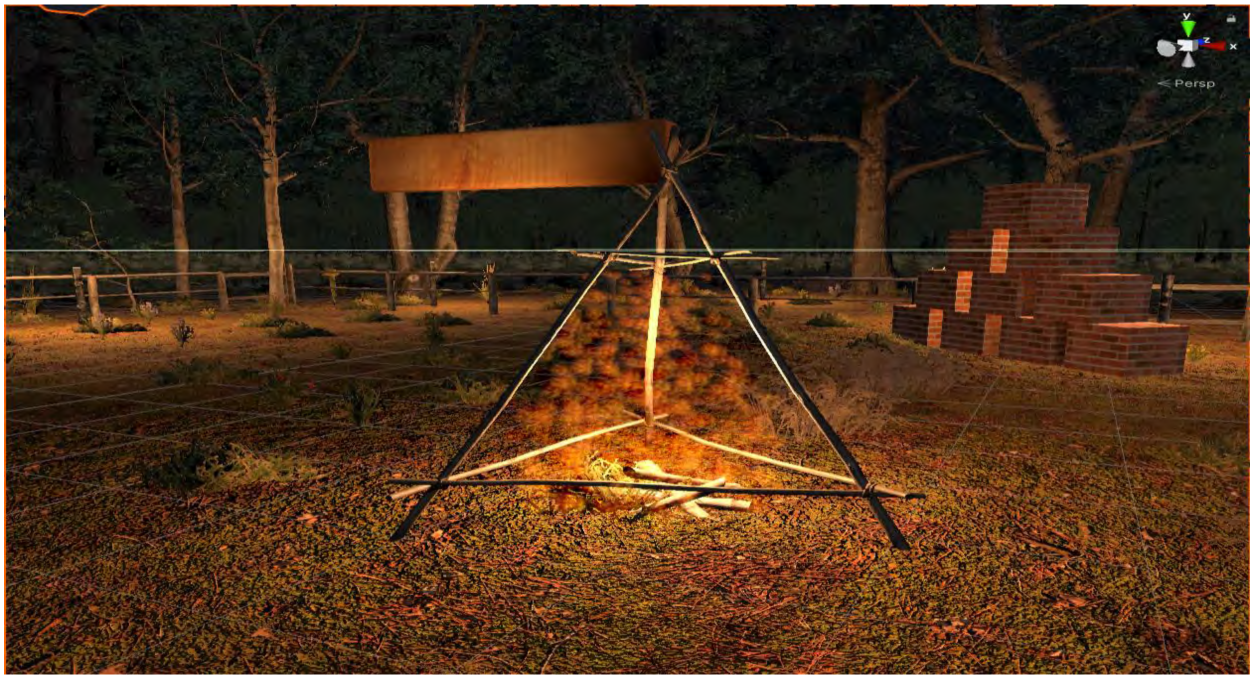


Figure 4-6 Fire using particle system

To bring dynamic animations to the application, I utilized Unity's Animation System. Unity's Animation System is a robust tool that allows developers to create, control, and blend animations for characters, objects, and other elements within the virtual environment. It provides a comprehensive set of features, including keyframe animation, blend trees, inverse kinematics (IK), and animation events.

Keyframe animation allows developers to define specific poses or transformations at different points in time, creating smooth and realistic motion. Blend trees enable the blending of multiple animations based on certain parameters, providing seamless transitions between different movements or states. Inverse kinematics (IK) allows for precise control of character limbs and joints, ensuring accurate interaction with the environment or other objects.

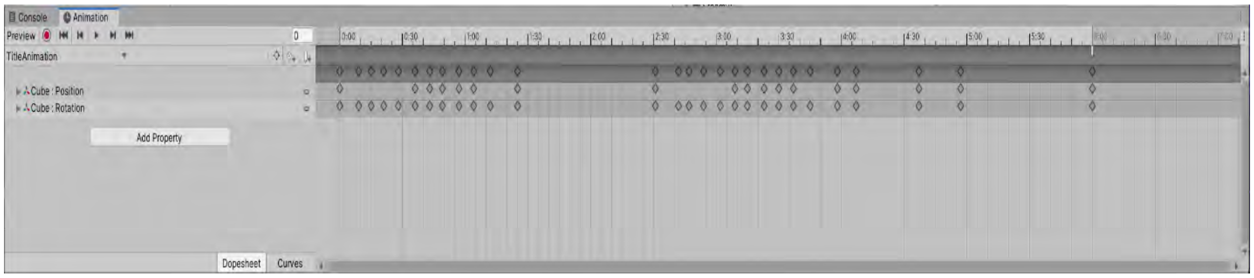


Figure 4-7 Unity's Keyframe Animation

Furthermore, Unity's Animation System supports animation events, which allow developers to trigger specific actions or behaviors at designated points during an animation sequence. This functionality enables synchronization with other game mechanics, audio cues, or visual effects, adding depth and interactivity to the animations.

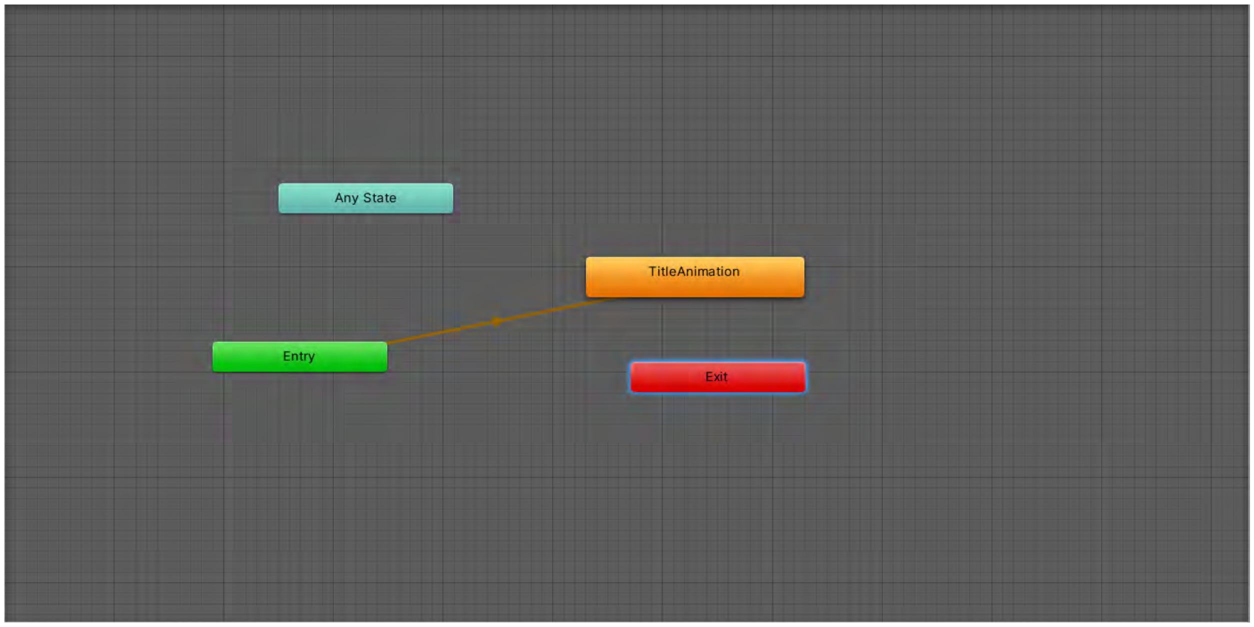


Figure 4-8 Animator

To achieve visually appealing graphics and optimize performance in the application, I employed Unity's Universal Render Pipeline (URP). Unity's URP is a highly customizable rendering solution that provides a balance between high-quality visuals and efficient rendering for a wide range of platforms, including PC, console, mobile, and VR.

The Universal Render Pipeline offers a set of features and optimizations designed to maximize the capabilities of modern hardware while maintaining flexibility and ease of use. It leverages the latest rendering techniques, such as physically-based shading, post-processing effects, and dynamic lighting, to create visually appealing and realistic scenes.

Additionally, URP supports various rendering features like GPU instancing, lightweight rendering passes, and shader variants stripping, which contribute to improved performance and faster rendering times. These optimizations are crucial for maintaining a high frame rate and providing a smooth experience to the user.

Moreover, URP integrates seamlessly with Unity's editor, providing a streamlined workflow for artists and developers. It offers an intuitive interface for configuring rendering settings, creating custom shaders, and fine-tuning visual effects. This accessibility empowers developers to achieve their desired visual aesthetic and efficiently iterate on the graphical aspects of the application.

4.3 Mechanics and Gameplay Development

The mechanics and gameplay of a game are vital for bringing it to life. They encompass all player actions, input handling, interactions, and dynamic features that shape the gameplay experience. To develop and implement these aspects, I utilized Unity's C# scripting capabilities. C# scripting in Unity allowed me to write the underlying logic and functionality that brings the application to life. By attaching C# scripts to game objects, I was able to define specific behaviors and functionalities based on the code within those scripts. This level of control enabled me to create dynamic and interactive elements within the application.

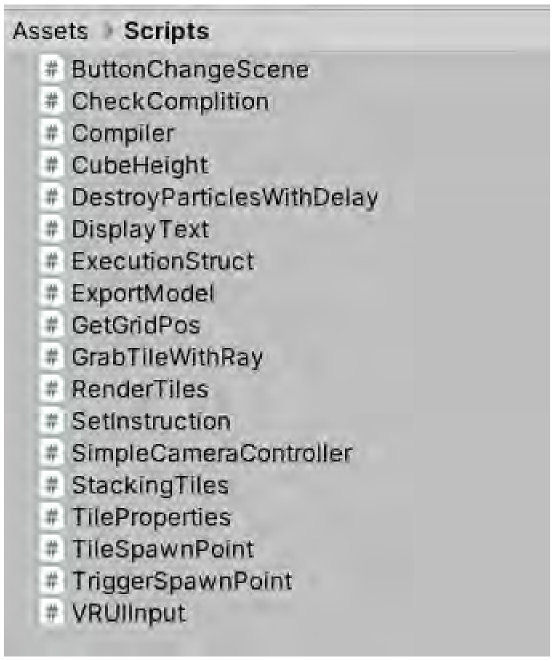
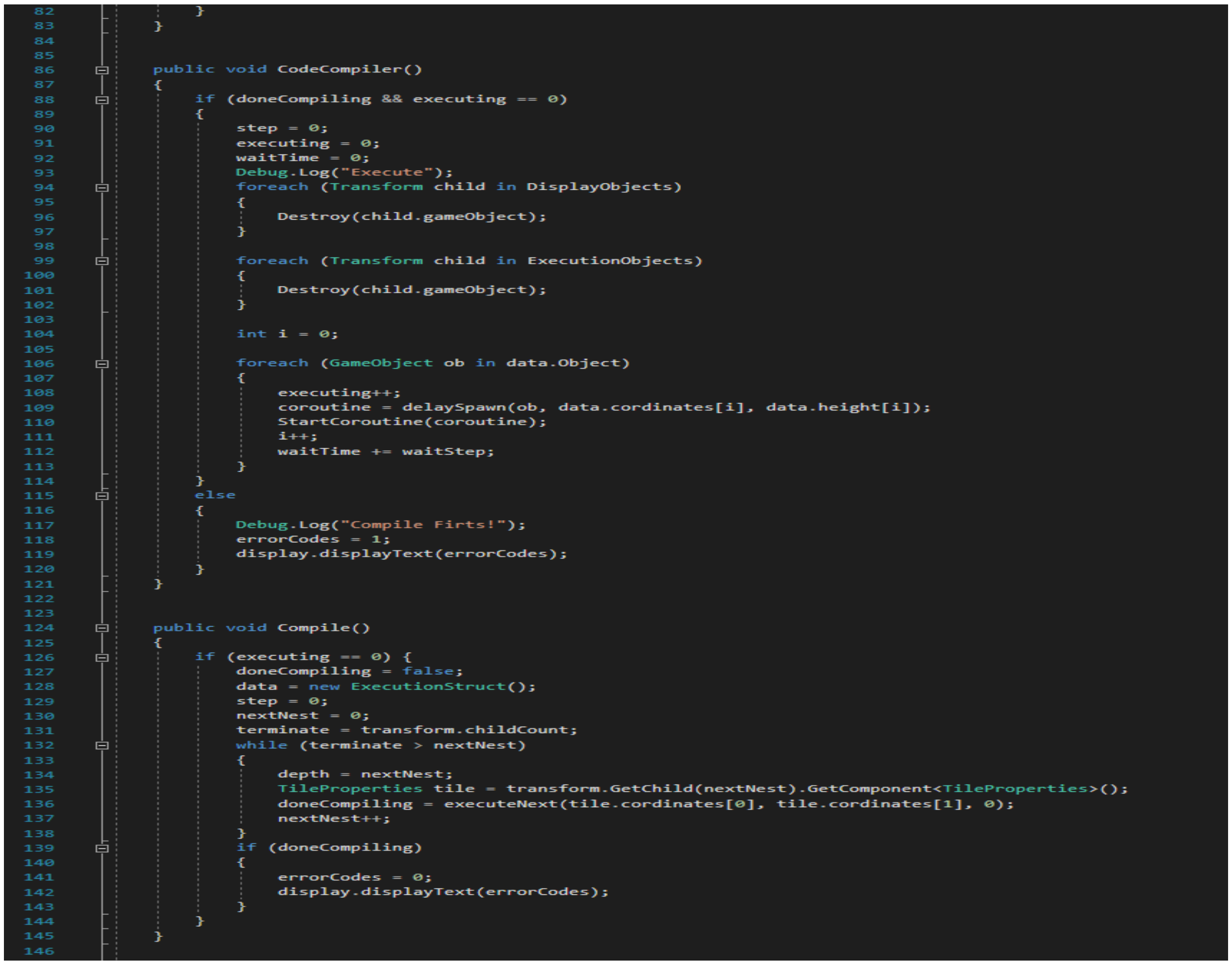


Figure 4-9 C# Scripts

Through C# scripting, I could handle player input, implement game rules, manage object interactions, and create dynamic features such as the creation of the dynamic grid, the compiler and many more. By leveraging the flexibility and versatility of C# scripting in Unity, I was able to tailor the mechanics and gameplay to suit the unique requirements of my application.



```
            }
        }


        public void CodeCompiler()
        {
            if (doneCompiling && executing == 0)
            {
                step = 0;
                executing = 0;
                waitTime = 0;
                Debug.Log("Execute");
                foreach (Transform child in DisplayObjects)
                {
                    Destroy(child.gameObject);
                }

                foreach (Transform child in ExecutionObjects)
                {
                    Destroy(child.gameObject);
                }

                int i = 0;

                foreach (GameObject ob in data.Object)
                {
                    executing++;
                    coroutine = delaySpawn(ob, data.cordinates[i], data.height[i]);
                    StartCoroutine(coroutine);
                    i++;
                    waitTime += waitStep;
                }
            }
            else
            {
                Debug.Log("Compile Firts!");
                errorCodes = 1;
                display.displayText(errorCodes);
            }
        }


        public void Compile()
        {
            if (executing == 0) {
                doneCompiling = false;
                data = new ExecutionStruct();
                step = 0;
                nextNest = 0;
                terminate = transform.childCount;
                while (terminate > nextNest)
                {
                    depth = nextNest;
                    TileProperties tile = transform.GetChild(nextNest).GetComponent<TileProperties>();
                    doneCompiling = executeNext(tile.cordinates[0], tile.cordinates[1], 0);
                    nextNest++;
                }
                if (doneCompiling)
                {
                    errorCodes = 0;
                    display.displayText(errorCodes);
                }
            }
        }

```

Figure 4-10 Part of application's Compiler in C#

4.4 Networking

To incorporate multiplayer functionality in my application and take advantage of 5G technology, I integrated the Photon Engine for Unity (PUN) for networking. By integrating PUN into my Unity project, I gained access to a set of powerful networking tools and libraries specifically designed for multiplayer experiences.

The networking process begins with the initialization of the PUN framework in my Unity project. This step establishes the necessary infrastructure for network connections and enables communication between players.

In the context of PUN, the first player who creates a room on a Photon PUN server becomes the host, while the subsequent players who join the room serve as clients. The distinction between the host and clients is primarily for networking communication purposes and does not impact the gameplay itself.

Once the players are connected and synchronized within the room, they can commence the multiplayer experience. PUN facilitates the synchronization of players' states and game objects across all connected clients, ensuring that everyone has a consistent view of the game world.

To share game objects between players, PUN provides a convenient method called PhotonNetwork.Instantiate. This method allows me to instantiate game objects that need to be shared across the network, ensuring that all players see and interact with the same objects in their respective game instances. In order for these game objects to be "network aware" and accessible across the network, the PhotonView script needs to be attached to them. The PhotonView script handles the serialization and transmission of the object's state, enabling synchronization with other clients.

Photon and its clients utilize a highly optimized binary protocol for efficient communication. This protocol is designed to be compact while remaining easily parseable. The specific details of the protocol implementation are abstracted and handled behind the scenes, relieving developers from needing to understand its inner workings.

The Photon binary protocols are organized into multiple layers. At the lowest level, UDP (User Datagram Protocol) is employed to transmit the messages sent by the clients. Within the standard UDP datagrams, various headers are used to incorporate the desired properties of Photon's functionality. These headers include optional reliability, sequencing, message aggregation, time synchronization, and other relevant features.

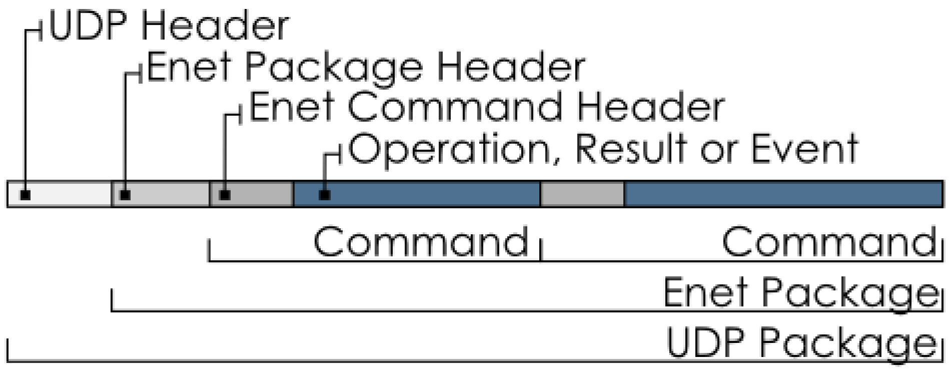


Figure 4-11 Photon's Binary Protocol

To elaborate, each UDP package consists of an eNet header and at least one eNet command. The eNet command encapsulates the messages exchanged between Photon and its clients, encompassing operations, results, or events. These commands are composed of an operation header, which provides context and metadata, as well as the associated data supplied by the sender. (photonengine, n.d.)

By utilizing these PUN functionalities, such as room creation, synchronization, shared object instantiation, I integrated multiplayer networking into my application. This integration enables players to connect, collaborate, and experience a shared virtual environment, harnessing the capabilities of 5G technology to enhance the multiplayer experience.

5 Conclusion and Future Work

In conclusion, this thesis has presented the design and implementation of a virtual reality (VR) application that aims to enhance the learning experience of coding concepts while leveraging the potential of 5G technology. Through the integration of VR and multiplayer functionality, the application provides an immersive and collaborative environment where users can learn and practice coding principles in real-time with other participants.

The technical aspects of the application were carefully considered and implemented. The Unity Game Engine served as the foundation, offering a robust set of features and tools for VR development. The application utilized the HTC VIVE headset, OpenVR SDK, and SteamVR plugin to enable seamless interaction with the VR hardware. Additionally, the Photon Engine facilitated the multiplayer functionality, allowing multiple users to participate in the same experience simultaneously.

The development process involved creating a virtual environment using Unity's built-in tools, such as the terrain tool for shaping the landscape and the ProBuilder tool for creating 3D objects. Visual effects were achieved using the Unity Particle System, while animations were implemented using Unity's animation system.

C# scripting played a crucial role in the implementation of gameplay mechanics, handling player input, managing object interactions, and creating dynamic features like the dynamic grid creation and the compiler functionality.

Furthermore, the application harnessed the capabilities of 5G technology to enable seamless multiplayer experiences. Through the low latency and high bandwidth offered by 5G networks, users could collaborate, share their coding creations, and observe each other's projects in real-time. This aspect of the application not only enhanced the learning experience but also fostered a sense of community and collaboration among participants.

The Unity Universal Render Pipeline (URP) was employed to achieve visually appealing and optimized graphics, ensuring a high-quality and immersive VR experience.

In summary, this thesis has successfully demonstrated the potential of VR technology, coupled with the capabilities of 5G networks, in revolutionizing coding education. By combining immersive experiences, interactive gameplay, multiplayer functionality, and the power of 5G, the application offers a unique and effective approach to learning coding concepts. The integration of these technologies opens up new possibilities for remote learning, collaboration, and real-time coding experiences.

Future work could focus on expanding the application's features and content, incorporating additional coding concepts and challenges, and further optimizing the performance and user experience within the context of 5G networks.

Overall, this thesis has provided valuable insights into the design, development, and implementation of a VR coding application that leverages the potential of 5G technology. The combination of immersive VR experiences, multiplayer functionality, and the power of 5G networks holds tremendous promise for the future of coding education, paving the way for innovative and collaborative learning environments.

6 References

[1] ossovr. (n.d.). Retrieved from ossovr.com: https://www.ossovr.com

[2] photonengine. (n.d.). *Binary Protocol*. Retrieved from doc.photonengine.com: https://doc.photonengine.com/pun/current/reference/binary-protocol

[3] techrow. (n.d.). Retrieved from techrow.com: https://www.techrow.org

[4] unity3d. (n.d.). *Stereo rendering*. Retrieved from docs.unity3d.com: https://docs.unity3d.com/Manual/SinglePassStereoRendering.html

[5] unity3d. (n.d.). *Terrain tools*. Retrieved from docs.unity3d.com: https://docs.unity3d.com/Manual/terrain-Tools.html