PROYECTO FIN DE GRADO

TÍTULO: Ampliación de un videojuego terapéutico para mejora del procesamiento auditivo: creación de un módulo de localización de sonidos en el espacio 3D

AUTOR/A: Moheng Li

TITULACIÓN: Grado en Ingeniería de Sonido e Imagen

DIRECTOR/A:

TUTOR/A: Martina Eckert

DEPARTAMENTO: Ingeniería Audiovisual y Comunicaciones

VºBº TUTOR/A

Miembros del Tribunal Calificador:

PRESIDENTE/A: Gregorio Rubio

TUTOR/A: Martina Eckert

SECRETARIO/A: Enrique Rendón

Fecha de lectura: 17 de Julio de2024

Calificación:

El Secretario/La Secretaria,

Resumen

El propósito de este proyecto de fin de grado es ampliar y complementar el videojuego terapéutico "El Planeta Sonoa", integrando la tecnología de realidad virtual (RV) para ofrecer un nuevo método de tratamiento para niños de 5 a 14 años diagnosticados con Desorden del Procesamiento Auditivo Central (DPAC). Esta innovación tecnológica propone nuevas posibilidades en el ámbito de la rehabilitación y terapia.

El DPAC es un trastorno del procesamiento auditivo, definido como un defecto en el procesamiento de información relacionada con los estímulos auditivos por el sistema nervioso central. A pesar de que la audición de los pacientes es normal, tienen dificultades para procesar sonidos en el cerebro. Esto comúnmente resulta en problemas para distinguir tipos de sonidos, entender el lenguaje y localizar la fuente y dirección del sonido en ambientes ruidosos. Por lo tanto, el DPAC tiene un impacto significativo en el aprendizaje y la vida cotidiana de los niños. La terapia principal para estos pacientes ha sido la corrección de habilidades directas, es decir, entrenamiento rehabilitador enfocado en tareas de procesamiento auditivo como el entrenamiento de la frecuencia, la intensidad, la escucha dicótica, y la localización de sonidos.

Con el avance de la tecnología de videojuegos en la última década, tanto los videojuegos comerciales como los juegos serios han encontrado aplicaciones crecientes en el campo médico, ayudando en la rehabilitación física, intervenciones de salud mental y rehabilitación cognitiva. Los videojuegos, con su carácter entretenido y alta interactividad, han demostrado ser efectivos en aumentar la motivación y participación, especialmente entre niños y adolescentes. "El Planeta Sonoa" es un juego de video terapéutico modular diseñado específicamente para pacientes con CAPD, gamificando los métodos terapéuticos principales para DPAC y ofreciendo módulos especializados para entrenar diversas tareas de procesamiento auditivo.

Este proyecto sirve como una expansión crucial y complemento de "El Planeta Sonoa", llenando un vacío en el entrenamiento de localización espacial de sonidos en 3D, una habilidad que implica la capacidad de identificar la dirección y distancia de una fuente sonora. La tecnología de RV, combinada con tecnología de audio espacial, se adapta perfectamente a las necesidades de este entrenamiento. Por lo tanto, utilizando estas tecnologías junto con el motor de juego, se ha diseñado y desarrollado un videojuego de RV para entrenar esta habilidad específica.

En el juego, se generan aleatoriamente objetos alrededor del jugador, de los cuales solo uno emite sonido. El jugador debe localizar y seleccionar correctamente el objeto sonoro para entrenar su habilidad de localización de sonido. Para adaptarse a diferentes pacientes en diversas etapas de tratamiento, el juego ofrece múltiples configuraciones ajustables, como el número de objetos, el alcance de la generación de objetos y el tipo de audio emitido por el objeto sonoro etc., proporcionando una personalización detallada del entrenamiento.

En términos técnicos, este proyecto implica el uso del motor de juegos Unity y la escritura de scripts en el lenguaje C#. Unity, con su desarrollo robusto a lo largo de los años, ofrece una amplia gama de funcionalidades y compatibilidad con múltiples plataformas, incluyendo soporte para diversos dispositivos de RV. Gracias a su versión gratuita y los abundantes recursos prefabricados disponibles en el Asset Store de Unity, optar por esta plataforma no solo es técnica sino también económicamente ventajoso.

Tras múltiples pruebas realizadas por varios individuos, los resultados preliminares indican que, en comparación con métodos tradicionales, los videojuegos aumentan significativamente el interés y la concentración en el entrenamiento, recibiendo evaluaciones positivas de los participantes. Sin embargo, debido a la pequeña muestra y la naturaleza subjetiva de las pruebas, además de que los participantes no eran pacientes con DPAC y no se realizaron bajo un diseño experimental controlado estrictamente, los resultados tienen una validez limitada. Se espera colaboraciones en el futuro con centros de tratamiento para realizar pruebas más profesionales.

Abstract

The purpose of this undergraduate thesis project is to expand and complement the therapeutic video game "El Planeta Sonoa" by integrating virtual reality (VR) technology, offering a new treatment method for children aged 5 to 14 diagnosed with Central Auditory Processing Disorder (CAPD). This technological innovation proposes new possibilities in the field of rehabilitation and therapy.

CAPD is an auditory processing disorder defined as a defect in the central nervous system's processing of information related to auditory stimuli. Although patients have normal hearing, they struggle to process sounds in the brain. This commonly results in difficulties distinguishing sound types, understanding language, and locating the source and direction of sounds in noisy environments. Thus, CAPD significantly impacts children's learning and daily life. The main therapy for these patients has been direct skill correction, namely rehabilitative training focused on auditory processing tasks such as frequency training, intensity, dichotic listening, and sound localization.

With advancements in video game technology over the last decade, both commercial video games and serious games have increasingly found applications in the medical field, aiding in physical rehabilitation, mental health interventions, and cognitive rehabilitation. Video games, with their entertaining nature and high interactivity, have proven effective in boosting motivation and engagement, especially among children and adolescents. "El Planeta Sonoa" is a modular therapeutic video game specifically designed for patients with CAPD, gamifying the main therapeutic methods for CAPD and offering specialized modules for training various auditory processing tasks.

This project serves as a crucial expansion and supplement to "El Planeta Sonoa," filling a gap in the training of 3D spatial sound localization, a skill that involves the ability to identify the direction and distance of a sound source. VR technology, combined with spatial audio technology, perfectly matches the requirements for this training. Therefore, using these technologies along with the game engine, a VR video game has been designed and developed to train this specific skill.

In the game, objects are randomly generated around the player, one of which emits sound. The player must locate and correctly select the sound-emitting object to train their sound localization ability. To accommodate different patients at various treatment stages, the game offers multiple adjustable settings, such as the number of objects, the range of object generation, and the type of audio emitted by the sound-emitting object, providing detailed customization of the training.

Technically, this project involves the use of the Unity game engine and scripting in the C# language. Unity, with its robust development over the years, offers a wide range of functionalities and multi-platform compatibility, including support for various VR devices.

Thanks to its free version and the plethora of prefabricated resources available on Unity's Asset Store, choosing this platform is both technically and economically advantageous.

After multiple tests conducted by various individuals, preliminary results indicate that, compared to traditional methods, video games significantly increase interest and concentration in training, with participants generally giving positive feedback. However, due to the small sample size and the subjective nature of the tests, as well as the fact that participants were not CAPD patients and the tests were not conducted under strictly controlled experimental designs, the results have limited validity. Future collaborations with treatment centers are anticipated to conduct more professional tests.

Índice de figuras

Lista de acrónimos

DPAC: Desorden del Procesamiento Auditivo Central

RV: Realidad Virtual

CITSEM: El Centro de Investigación en Tecnologías Software y Sistemas Multimedia para la Sostenibilidad

GAMMA: Grupo de Aplicaciones Multimedia y Acústica

MRTK: Mixed Reality Toolkit

HRTF: Head-related transfer function (Función de transferencia relacionada con la cabeza)

ISFE: Interactive Software Federation of Europe

RCT: Randomized controlled trial (Ensayo controlado aleatorio)

Índice de contenidos

1 Introducción

1.1 Marco y motivación del proyecto

En las últimas décadas, los videojuegos a menudo han sido considerados como una influencia negativa, con numerosos estudios y reportes que resaltan sus impactos adversos en los seres humanos, especialmente en niños y adolescentes. Sin embargo, con la llegada de la era digital, la tecnología ha experimentado un desarrollo extraordinario en los últimos años. Todas las tecnologías, incluyendo la de los videojuegos, han sido ampliamente extendidas y utilizadas en diversos campos. Los videojuegos están emergiendo como herramientas educativas y terapéuticas efectivas. Estos videojuegos, cuyo principal propósito no es el entretenimiento, se conocen como juegos serios, y su aplicación e investigación están en constante evolución.

Gracias a sus características de entretenimiento e interactividad, los videojuegos se han convertido en una nueva tendencia en el campo de la terapia y rehabilitación, especialmente en los juegos de entrenamiento para niños. A diferencia de los métodos de rehabilitación tradicionales que son aburridos y monótonos, los videojuegos ofrecen a los pacientes una experiencia completamente nueva. Sin embargo, como una tecnología relativamente nueva, su popularidad no es muy alta. En muchos campos de rehabilitación que necesitan estos juegos serios, aún no existen tales videojuegos, y la investigación sobre su efectividad es limitada. Por lo tanto, el uso de videojuegos como herramientas terapéuticas es una dirección de investigación con gran potencial y muchos beneficios sociales.

“El planeta Sonoa” es un videojuego terapéutico desarrollado en este contexto, dirigido a niños con Trastorno del Procesamiento Auditivo Central (DPAC). A través de un desarrollo modular, integra cinco juegos destinados a entrenar diferentes habilidades auditivas. Actualmente, este proyecto carece de un módulo para entrenar la capacidad de los pacientes para localizar sonidos en un espacio 3D.

El objetivo de este proyecto es ayudar a “El planeta Sonoa” a llenar este vacío en el entrenamiento de la capacidad de localización de sonidos. Dado que este entrenamiento implica la implementación de tecnología de audio espacial, aunque también se puede realizar en videojuegos basados en computadoras, estas no ofrecen la misma calidad de interacción multimodal que la tecnología de Realidad Virtual (RV). Los videojuegos en dispositivos de RV llevan la interacción visual y auditiva a un nivel óptimo. Por ello, este proyecto optó por desarrollar un videojuego terapéutico en RV.

1.2 Objetivos técnicos y académicos

Los objetivos de este proyecto fin de carrera son, desde el punto de vista técnico:

- Desarrollar un videojuego terapéutico en RV para el proyecto “El planeta Sonoa”, enfocado en entrenar la capacidad de localización de sonidos.
- Diseñar mecánicas de juego interesantes, escenas inmersivas y personajes atractivos para aumentar la diversión y la interactividad del juego, así como la atención de los pacientes.
- Asegurar la adaptabilidad del juego mediante el diseño de varios parámetros personalizables que ofrezcan diferentes niveles de dificultad para diferentes situaciones de los pacientes.
- Desarrollar una función de salida de resultados del juego que proporcione a los terapeutas una base para evaluar el efecto del tratamiento y monitorear el progreso de los pacientes.
- Diseñar pruebas rigurosas y razonables para demostrar la efectividad del diseño.

Desde el punto de vista académico, el proyectista adquiere las siguientes competencias y habilidades:

- Integrar conocimientos multidisciplinarios durante el proceso de desarrollo del proyecto, aplicando conocimientos de diseño de juegos, programación, ingeniería acústica, entre otros, en un proyecto práctico.
- Adquirir habilidades para distribuir el tiempo razonablemente, establecer planes y rastrear el progreso para asegurar la finalización oportuna del proyecto.
- Dado que el proyecto implica el desarrollo de un proyecto de RV en el motor Unity, mejorar las habilidades de programación en Unity y C#, así como aprender métodos de desarrollo para proyectos de RV.
- Aprender diseño estético, incluyendo el diseño de modelos y escenarios, así como cómo diseñar e implementar interfaces y interacciones amigables para el usuario, mejorando la jugabilidad y la experiencia del usuario.
- Desarrollar habilidades de trabajo en equipo, colaborando y comunicándose con los compañeros del equipo de desarrollo y el tutor, mejorando el proyecto a través de una comunicación efectiva.
- Desarrollar habilidades para resolver problemas cuando surjan diversas dificultades en el proyecto.

1.3 Estructura del resto de la memoria

Capítulo 2: Marco tecnológico: En este capítulo se detallará todos los campos tecnológicos involucrados en el proyecto, incluyendo conocimientos sobre DPAC, juegos serios, audio espacial, entre otros. Además, se presentará una introducción a las herramientas técnicas utilizadas como Unity y Blender.

Capítulo 3: Especificaciones y restricciones de diseño: Esta sección enumerará los principales problemas a resolver en la solución del proyecto, así como los requisitos y restricciones de diseño de la solución propuesta.

Capítulo 4: Descripción de la solución propuesta: Se describirá el diseño y la implementación específicos del proyecto, desde el diseño conceptual del juego hasta los métodos específicos de implementación de las mecánicas del juego.

Capítulo 5: Resultados: Se mostrarán los procesos de prueba del proyecto y el análisis de los resultados obtenidos.

Capítulo 6: Presupuesto: Se estimará el costo del proyecto, analizando la composición de los costos del proyecto y la razonabilidad de estos costos.

Capítulo 7: Impacto del proyecto: Se analizará el impacto del proyecto en el medio ambiente, la sociedad y el ámbito académico.

Capítulo 8: Conclusiones: Se realizará una revisión de la implementación del proyecto, resumiendo los logros y las deficiencias del proyecto.

Capítulo 9: Trabajos futuros: Se analizarán las posibles extensiones del proyecto en el futuro, basándose en las deficiencias y el potencial identificado durante el desarrollo del proyecto.

2 Marco tecnológico

El conocimiento y la comprensión adecuados de los campos y tecnologías relacionados con el proyecto son la base para la planificación y el diseño del proyecto. Este proyecto implica el uso del motor de juegos Unity y el lenguaje de programación C# para crear un videojuego terapéutico en RV para niños con DPAC. El principal campo de este proyecto es el de los juegos serios. Debido a que el contenido principal del juego es el entrenamiento de la capacidad de localización de sonidos en un espacio 3D, durante su implementación también se utilizará audio espacial. Además, el diseño estético es indispensable en los videojuegos, y en este proceso se utilizó principalmente el software de gráficos por computadora Blender para diseñar algunos modelos y escenarios.

El diseño de este proyecto se basa en un videojuego llamado “El Planeta Sonoa”. Este juego fue iniciado por la profesora Martina Eckert y desarrollado por el Grupo de investigación de Aplicaciones MultiMedia y Acústica (GAMMA), un grupo oficial certificado por la Universidad Politécnica de Madrid (UPM) y actualmente perteneciente al Centro de Investigación en Tecnologías Software y Sistemas Multimedia para la Sostenibilidad (CITSEM) [1]. Uno de los actuales enfoques de investigación de este grupo es la gamificación para la rehabilitación, con el objetivo de mejorar los juegos serios. Antes de este proyecto, “El Planeta Sonoa” ya había integrado cinco módulos, cada uno desarrollado por estudiantes de la UPM como proyectos de fin de grado, diseñados para entrenar diferentes habilidades auditivas. La creación de este proyecto llena el vacío en el entrenamiento de la capacidad de localización de sonidos.

En resumen, para llevar a cabo este proyecto, es fundamental comprender los conocimientos relacionados con DPAC, juegos serios, audio espacial y RV, así como dominar herramientas como el motor de juegos, el lenguaje C# y Blender.

2.1 Desorden del procesamiento auditivo central (DPAC)

2.1.1 Patología y síntomas de la DPAC

El sistema auditivo es un sistema complejo que involucra ondas mecánicas y procesamiento neurobiológico. El sonido llega al tímpano en forma de ondas mecánicas, se transmite y convierte en señales eléctricas a través del oído medio e interno. Posteriormente, estas señales eléctricas son transmitidas por el nervio auditivo al centro cerebral para un procesamiento y análisis adicionales, lo que permite a las personas comprender e interpretar la información contenida en el sonido [2]. En este sistema, cualquier problema en cualquier etapa puede llevar a una disminución de la función auditiva. Los niños con DCAP tienen problemas en el nervio auditivo central (trastorno del desarrollo neurológico, retraso en la maduración, enfermedades del sistema nervioso, etc.), lo que provoca dificultades en el procesamiento de la información. Aunque tienen una capacidad normal para detectar sonidos, les resulta difícil identificar y distinguir sonidos. El síntoma más común es la incapacidad para

comprender el habla en entornos ruidosos [3]. Además, dado que la audición es insustituible para el aprendizaje de habilidades como el habla, la lectura, la música, etc., las deficiencias en las habilidades auditivas causadas por el DCAP también generan numerosos problemas en el crecimiento y aprendizaje de los niños. La Tabla 1, tomada del artículo de Óscar Cañete, resume en detalle las diferentes habilidades auditivas [4], lo que permite comprender mejor el impacto del DCAP en los niños. "El Planeta Sonoa" ha diseñado módulos de entrenamiento específicos para diferentes habilidades basándose en esta tabla.

Tabla 1 Habilidades auditivas importantes en el proceso de aprendizaje [4].

Habilidad	Descripción
Discriminación	Diferenciación de sonidos de diferente frecuencia, duración o intensidad.
Localización	Ubicación de la fuente sonora.
Discriminación fonológica	Diferenciación de los elementos fonémicos del habla que son acústicamente similares.
Encierro (Closure) auditivo	Compresión de un mensaje o palabra “completo” cuando una porción está ausente.
Separación auditiva en ruido	Identificación del hablante primario en presencia de ruido de fondo.
Asociación auditiva	Capacidad e otorgar un significado a las palabras.
Memoria auditiva	Capacidad para almacenar y recordar estímulos en orden o secuencia apropiada.
Atención auditiva	Capacidad de dirigir y mantener la atención hacia una señal acústica relevante por un periodo apropiado de tiempo.

2.1.2 Métodos diagnósticos y terapéuticos tradicionales

El diagnóstico de DPAC incluye, además de la evaluación de la historia clínica, evaluación audiológica y evaluación del lenguaje, la realización de evaluaciones médicas específicas, que son el método principal. Existen muchas pruebas para DPAC, que se dividen principalmente en pruebas conductuales y pruebas electrofisiológicas. Las pruebas conductuales incluyen: pruebas de estímulo binaural, pruebas de reducción de redundancia y pruebas de procesamiento temporal, entre otras. Las pruebas electrofisiológicas implican la estimulación de la corteza cerebral a través de numerosos dispositivos. Comparativamente, las pruebas conductuales son más preferidas debido a su facilidad de operación [4]. Además, actualmente existen muchas pruebas estandarizadas de procesamiento auditivo, como la prueba SCAN desarrollada en Estados Unidos, que también se utiliza a menudo para el diagnóstico de DCAP [3]. Esta incluye: (1) Identificación de palabras monaurales en ambientes ruidosos, (2) Identificación de palabra suelta acústicamente degradada, (3) Test de escucha dicótica y (4) Test de frases [5].

Los resultados de la investigación sobre la neuroplasticidad muestran que la plasticidad y la función nerviosa pueden mejorarse mediante la estimulación repetida, especialmente en niños y adolescentes [4]. Esto sienta las bases teóricas para el tratamiento de rehabilitación de DCAP. Los métodos de tratamiento tradicionales actuales se centran principalmente en el entrenamiento auditivo, que incluye, entre otros: discriminación auditiva, discriminación de fonemas, audición temporal, entrenamiento binaural, localización y lateralización del sonido [6]. Las características de estos entrenamientos los hacen muy adecuados para realizarlos en una computadora. Actualmente, el uso de programas de software para el entrenamiento auditivo y la gamificación del entrenamiento auditivo es cada vez más común [7].

2.2 Videojuegos serios

Una característica esencial de los videojuegos, que los diferencia de otros medios como películas y televisión, es su alta interactividad. Permiten la participación del jugador en todo el proceso del sistema, proporcionando a continuación retroalimentación adecuada [8]. Los videojuegos son hoy en día una actividad de entretenimiento esencial para casi todos los niños y adolescentes. Un informe de estadísticas relacionadas con videojuegos en Europa del año 2022, publicado por la Interactive Software Federation of Europe (ISFE), muestra que el 71% de los niños europeos de 6 a 10 años juegan videojuegos, y este porcentaje aumenta al 81% en el grupo de adolescentes de 11 a 14 años [9]. El impacto de los videojuegos en la salud física y mental de los adolescentes ha sido siempre un tema controvertido. Aunque los efectos negativos relacionados con la violencia y la adicción han sido discutidos en múltiples ocasiones, existen evidencias que muestran que los videojuegos también tienen un lado positivo [8], especialmente en la promoción de diversas habilidades cognitivas como la atención [10], el aumento de la motivación, la mejora del control emocional [11] y el desarrollo de habilidades sociales en estos cuatro ámbitos principales [12].

Debido a los numerosos beneficios de los videojuegos, surgió el concepto de juegos serios, propuesto inicialmente por Clark C. Abt en su libro de 1970, "Serious Games" [13]. La definición de juegos serios varía ampliamente, pero una definición común es "cualquier forma de videojuego que no tenga el entretenimiento como su objetivo principal y que puede ser jugado por cualquier número de personas en cualquier plataforma" [14]. Sus principales áreas de aplicación incluyen: educación y capacitación, mejora del bienestar, patrimonio cultural, y biomedicina y cuidado de la salud [15].

La comunidad médica también ha reconocido el gran potencial de los juegos serios, comenzando a gamificar aspectos como el monitoreo de la salud, intervenciones de tratamiento y entrenamiento de rehabilitación[15], [16]. Hasta la fecha, se han desarrollado muchos juegos serios exitosos que han logrado resultados significativos en motivar a los pacientes y mejorar su estado de salud. Pamela M. Kato, en su artículo de revisión "Video Games in Health Care: Closing the Gap", resume y explica con cierto detalle los logros alcanzados en este campo [17].

2.2.1 Situación actual de los videojuegos terapéuticos

El alcance de los juegos serios es muy amplio. El videojuego terapéutico abordado en este proyecto es solo una rama, es decir, su aplicación en la rehabilitación médica. Dentro de esta rama, existen muchas subramas. Según la clasificación de juegos serios terapéuticos propuesta por Rego P, en la Tabla 2 se ha organizado una posible clasificación de los juegos terapéuticos existentes. Según el nivel de patología, se pueden dividir en dos categorías: rehabilitación cognitiva y rehabilitación de habilidades motoras. Además, también se pueden clasificar de manera más detallada según el método de interacción del juego, el tipo y si tienen o no una función de ajuste dinámico de la dificultad (Adaptabilidad) [18].

Tabla 2: Taxonomía de los juegos serios terapéuticos [18].

Ámbitos de aplicación	Tecnología interactiva	Interfaz de juego	Número de jugadores	Tipo de juego	Más criterio de clasificación
Rehabilitación cognitiva: trastornos del neurodesarrollo, psicosis depresivas, trastornos de ansiedad, psicosis asociadas a traumatismos y factores estresantes, trastornos neurocognitivos, etc.	Ratones, teclados, pantallas montadas en la cabeza (HMD), interfaces hápticas como guantes de datos, dispositivos de seguimiento del movimiento	2D; 3D	Único; Multi	Juegos para evaluar el movimiento (prensión, alcance y agarre); Juegos de simulación; Juego de estrategia; Mixto	Adaptación (Sí/No); Evaluación del rendimiento (sí/no); Seguimiento del progreso; Portabilidad del juego
Rehabilitación deportiva: rehabilitación de accidentes cerebrovasculares, entrenamiento del equilibrio, lesión cerebral adquirida, movilidad en silla de ruedas, enfermedad de Parkinson, rehabilitación ortopédica, etc.					

En comparación con los métodos tradicionales de rehabilitación, que son repetitivos y aburridos [19], el uso de juegos serios para la rehabilitación tiene una gran ventaja en cuanto a aumentar la motivación del paciente para entrenar. Debido a las características de diversión e interactividad inherentes a los juegos, se mejora la sensación de frustración que la inevitable repetición en la rehabilitación causa a los pacientes [20]. El diseño del juego, con una dificultad que aumenta gradualmente, permite que los pacientes experimenten constantemente la sensación de logro que traen los desafíos, al mismo tiempo que mejora sus habilidades relevantes. Además, la concentración que los juegos proporcionan a los pacientes también desvía su atención del dolor físico durante el entrenamiento [19].

El notable efecto de los juegos serios en el campo de la salud no solo tiene un respaldo teórico, sino que muchas instituciones de investigación en campos relacionados también han realizado

evaluaciones sistemáticas de sus efectos reales. Por ejemplo, en un estudio liderado por el Departamento de Medicina de la Universidad de Pittsburgh, Pensilvania, en 2010, se seleccionaron 38 estudios de 1452 artículos sobre la aplicación de juegos terapéuticos que utilizaron estrictos ensayos controlados aleatorios (RCT). Cada estudio evaluó la capacidad de los juegos terapéuticos para promover la salud y/o mejorar enfermedades. Según los criterios de este estudio, si el efecto de la intervención de los juegos terapéuticos era superior al del grupo de control (rehabilitación tradicional), el resultado se consideraba positivo. Entre los proyectos de rehabilitación relacionados con la fisioterapia, la psicoterapia y la transferencia del dolor, el 67%-100% logró sus objetivos, produciendo resultados positivos [21].

Un ejemplo representativo de un juego serio terapéutico es Re-Mission, un videojuego diseñado para niños con cáncer. En el juego, los jugadores toman el papel de un nanorobot que ataca las células cancerosas y controla obstáculos comunes en el tratamiento del cáncer, como náuseas y estreñimiento. Tras ensayos RCT en 34 centros de tratamiento internacionales, se concluyó que los niños que jugaron este juego mostraron una mejora significativa en el cumplimiento de los planes de tratamiento y en su motivación para el tratamiento [22]. Actualmente, este juego se considera una herramienta de tratamiento exitosa y se distribuye a más pacientes.

Otro ejemplo representativo de un juego terapéutico es el sistema "Blexer", desarrollado en colaboración por múltiples instituciones, incluido CITSEM, para personas con discapacidades motoras. El sistema admite varios sensores de movimiento, principalmente Kinect. Su característica más destacada es que no se limita a desarrollar minijuegos para patologías específicas, sino que aumenta la inmersión a través de la experiencia de un "juego completo" y ofrece la capacidad de ajustar remotamente la dificultad y configuración del juego a través de la plataforma del sistema "Blexer-med". Esto proporciona a los pacientes una experiencia de juego de rehabilitación completa y personalizada. Actualmente, el primer juego de aventura completo en la plataforma, "Phiby's Adventure", está siendo probado por voluntarios con diversas enfermedades del sistema nervioso [23].

2.2.2 Los videojuegos terapéuticos relacionados con el entrenamiento auditivo

Según los resultados de búsqueda en bases de datos académicas y no académicas como ScienceDirect, Google Scholar, ResearchGate, APA y Google, en comparación con otras áreas de la salud, actualmente no hay muchos juegos serios terapéuticos específicos para DPAC, pero aún existen algunos juegos representativos que han logrado resultados sustanciales en el entrenamiento auditivo. Comprender las ventajas y limitaciones de estos juegos es de gran ayuda para el desarrollo de este proyecto.

“Fast ForWord”: Aunque el propósito de este juego serio es tratar a niños con dislexia, su método de entrenamiento de plasticidad neuronal, que consiste en establecer correspondencias entre sonidos y letras o palabras, también beneficia a los niños con DPAC.

Evidencias preliminares de varios estudios sugieren que este juego puede mejorar el procesamiento auditivo [24].

"Earobics": Este es un juego serio más centrado en el entrenamiento de la capacidad de procesamiento auditivo. A través de diferentes minijuegos, como la identificación de sonidos en distintos entornos acústicos, la ordenación de sonidos, la vinculación de sonidos con letras y la memoria de palabras sonoras, mejora las habilidades de conciencia fonológica y de lectura y escritura en niños con trastornos del lenguaje, el aprendizaje y la lectura. Sin embargo, tres estudios sobre la efectividad de este juego muestran que puede tener un impacto positivo en la capacidad de procesamiento auditivo de los niños, pero no hay evidencia efectiva que respalde la mejora en las habilidades de lectura y escritura [24].

"Sound Storm" (LiSN & LEARN): Un juego para niños de 6 a 12 años con trastorno del procesamiento espacial (una forma de DPAC que se manifiesta como incapacidad para localizar fuentes de sonido). La mecánica principal del juego es elegir la respuesta correcta de entre cinco imágenes basándose en la pregunta escuchada. A través de la modificación de la diferencia de tiempo y la diferencia de intensidad entre los canales de los oídos (usando auriculares) se crea una sensación de espacio sonoro, y se aumenta el desafío ajustando el volumen de las preguntas y los sonidos interferentes [25]. Aunque el contenido del juego es algo monótono, tiene ventajas destacadas, como un excelente diseño artístico y una función profesional de salida de resultados. Al rastrear el progreso del jugador en la capacidad de resistencia al ruido, evalúa dinámicamente el estado de entrenamiento del jugador. Un informe de evaluación de la efectividad de este juego, basado en datos recopilados de 38 centros de audición, señaló que este juego tiene un efecto significativo en la intervención y tratamiento de niños con DPAC [26]. La figura 1 muestra las capturas de pantalla del juego facilitadas por la web oficial del juego y el informe de resultados resultante [25].

Figura 1: Interfaz del juego terapéutico "Sound Storm" (izquierda) y su interfaz de informe de resultados (derecha).

2.2.3 “El Planeta Sonoa”

Comprender y analizar los síntomas de los pacientes con DPAC y el estado actual de los juegos serios terapéuticos relacionados, es fundamental para realizar un análisis detallado del juego principal al que pertenece este proyecto, "El Planeta Sonoa", con el fin de determinar el estilo y la dirección del diseño del juego a desarrollar. Como se mencionó al principio de este capítulo, "El Planeta Sonoa" es un juego terapéutico serio de múltiples módulos desarrollado para niños con DPAC. La característica de tener múltiples módulos es lo que lo diferencia de otros juegos mencionados anteriormente. Basándose en las habilidades auditivas importantes resumidas en el artículo de Óscar Cañete (Tabla 2) [4], se han desarrollado módulos de juego correspondientes a diferentes habilidades. Para mejorar la inmersión del jugador, el juego ha diseñado un mundo coherente, en el que cada módulo corresponde a un continente diferente de este planeta. En cada continente hay diferentes animales parlantes que asignan diversas tareas (entrenamiento auditivo) a los jugadores, quienes desempeñan el papel de viajeros que llegan a este planeta en una nave espacial y se dirigen a diferentes continentes para jugar según sus necesidades de entrenamiento.

Desde el inicio del proyecto, "El Planeta Sonoa" ha pasado por dos grandes fases de desarrollo. En la primera fase inicial, se desarrollaron dos módulos:

- **“Polaris”**: Creado por Luna Sampedro Jiménez y perfeccionado por Esther García Medina, este módulo se centra en entrenar la memoria auditiva y la atención auditiva de los jugadores. En un entorno similar a la Antártida, los jugadores deben elegir la opción correcta basada en los sonidos y gestos emitidos por los pingüinos en presencia de ruido de fondo [27], [28].
- **“Selvalia”**: Desarrollado por María Cristina Ojeda Egocheaga, este módulo se centra en entrenar la capacidad de discriminación, es decir, la capacidad de distinguir y diferenciar elementos fonéticos similares. En un entorno selvático, los jugadores deben encontrar los objetos correspondientes basándose en los sonidos emitidos por los monos [29].

En la segunda fase, que comenzó en 2022, se diseñaron y desarrollaron cuatro módulos, incluido este proyecto. Los otros tres módulos se completaron en julio de 2023, y son los siguientes:

- **“Cavernia”**: Desarrollado por Bogdan Morar, este módulo tiene como objetivo entrenar la asociación y la atención auditivas de los jugadores. En una cueva, los jugadores escuchan historias contadas por un oso y deben responder a una serie de preguntas basadas en el contenido de las historias. Aunque el módulo solo tiene un número fijo de historias pregrabadas, la integración de IA generativa permite generar diferentes preguntas y respuestas, garantizando la repetibilidad del juego [30].
- **“Kobarag”**: Desarrollado por Mario Ortega García, este módulo se centra en el entrenamiento de la memoria auditiva y la capacidad de discriminación de los jugadores (distinguir tonos de diferentes frecuencias y duraciones). En un entorno

desértico, los jugadores hacen que las serpientes de cascabel bailen al son de una flauta repitiendo cadenas de tonos largos o cortos, graves o agudos [31].

- **"Tropikus"**: Desarrollado por Pablo Lucas Olivares, este módulo se centra en la capacidad de escucha dicótica de los jugadores. En una isla tropical, los jugadores deben recibir diferentes estímulos en cada oído, escuchar diferentes palabras y encontrar los patrones correspondientes en la isla [32].

Dado que este proyecto involucra dispositivos de RV, mientras que las plataformas de los cinco proyectos completados están basadas solo en computadoras, después de su finalización, estos cinco proyectos se integraron en un software de videojuego, "El Planeta Sonoa". Uno de los objetivos de esta integración, y su mayor ventaja, es que a través del sitio web http://sonoa.citsem.upm.es/ y una base de datos, se implementa una función de gestión de usuarios para pacientes y terapeutas. En este sitio web, los terapeutas pueden configurar la dificultad y el volumen de los cinco juegos según las necesidades específicas de los pacientes, y pueden guardar los resultados de los juegos de los pacientes en la base de datos para su análisis, monitoreo y ajuste de los planes de tratamiento posteriores. Esta función fue desarrollada por el experto en desarrollo de aplicaciones del grupo GAMMA, Miguel Jiménez Arévalo.

Además, el diseño de todos los módulos de juego, incluido este proyecto, se llevó a cabo tras múltiples discusiones y colaboraciones con varios audiólogos del centro de tratamiento AUREA TAV [33]. Los terapeutas proporcionaron valiosas sugerencias sobre cómo diseñar parámetros para ajustar la dificultad del juego y qué tipos de entrenamientos y audios son más efectivos para habilidades específicas. Recientemente, el grupo Gamma también validó la efectividad del proyecto en la detección y diagnóstico de niños con DPAC mediante la realización de estrictos ensayos RCT. Tras evaluar las capacidades auditivas de 87 niños en edad escolar utilizando "El Planeta Sonoa", se concluyó que el proyecto parece ser confiable para el diagnóstico de DPAC [34]. Estos dos puntos confirman la profesionalidad y efectividad del proyecto, así como su amplio potencial futuro.

2.3 Teoría y desarrollo de la tecnología de sonido 3D

2.3.1 Principios de la localización acústico y Función de transferencia relacionada con la cabeza (HRTF)

El sistema auditivo humano localiza el sonido mediante un proceso complejo que depende principalmente de las señales recibidas por ambos oídos. Cuando un sonido llega a los oídos desde una determinada dirección, se generan diferentes señales de presión sonora en cada oído. Estas diferencias se deben principalmente a la diferencia de tiempo de llegada del sonido y a la diferencia de intensidad: Interaural Time Difference (ITD) y Interaural Level Difference (ILD). Además, la forma y estructura del pabellón auricular provocan efectos de reflexión, difracción y resonancia en el sonido, actuando como un filtro lineal que, según la dirección y

distancia de la fuente sonora, filtra las ondas sonoras y codifica la información espacial del sonido [35].

El ITD y el ILD, así como otra información clave para la localización del sonido, pueden obtenerse mediante la medición y el cálculo de la HRTF (Head-Related Transfer Function). La HRTF describe los cambios en la señal sonora desde la fuente hasta los oídos del oyente, causados por la reflexión y difracción de la cabeza, el pabellón auricular y el torso. En 1994, el MIT publicó el proceso y los resultados de la medición de la HRTF utilizando un micrófono de cabeza artificial KEMAR. El modelo KEMAR calcula la HRTF registrando las señales sonoras de diferentes direcciones con micrófonos colocados en ambos oídos. Una vez obtenida la HRTF, se puede extraer el ITD, el ILD y otra información necesaria para lograr la localización y renderización espacial del sonido [35], [36].

2.3.2 El proyecto de Oier

Para implementar el entrenamiento de localización del sonido en el juego, el motor del juego debe soportar la función de audio espacial. El motor Unity proporciona funcionalidades básicas de sonido 3D, pero para obtener un efecto de audio espacial más avanzado, este proyecto utilizó un plugin de audio espacial para Unity, diseñado y desarrollado por Oier Lauzirika Zarrabeitia, un ex estudiante de la UPM, en un trabajo académico. Oier utilizó los datos de medición de la HRTF del KEMAR del MIT mencionados anteriormente, los complementó con métodos de interpolación y desarrolló, con Visual Studio, bibliotecas DSP y binaurales para convertir los datos de la HRTF en señales binaurales, logrando así la simulación de sonido en un espacio tridimensional [37].

2.4 Los sistemas y softwares utilizados

2.4.1 El motor de video juego Unity

La tecnología de motores de juego es la tecnología principal para el desarrollo de juegos en la actualidad. Es un marco de software que integra diferentes motores como gráficos, física, sonido y scripting, proporcionando herramientas básicas para el desarrollo y producción de videojuegos, y simplificando la complejidad del desarrollo de juegos. Los motores de juego más utilizados actualmente incluyen Unity, Unreal Engine y Godot Engine. La ventaja de Unity radica en su compatibilidad con múltiples plataformas, especialmente con varios dispositivos de RV, lo que lo hace adecuado para el entorno de desarrollo de RV de este proyecto. Además, cuenta con una tienda de recursos rica y una comunidad de usuarios activa, lo que puede ahorrar costos económicos y de tiempo en el desarrollo [38].

En este proyecto, se eligió la versión 2021.3.18f1 de Unity para garantizar una mejor compatibilidad y mayor estabilidad con los proyectos anteriores [39].

2.4.2 Tecnología RV y Mixed Reality Toolkit (MRTK)

Como las gafas de RV de CITSEM son del modelo Explore de la marca Lenovo, es necesario construir el entorno adecuado para el dispositivo durante la fase de desarrollo del proyecto. El uso de este dispositivo requiere la descarga de la plataforma Windows Mixed Reality (WMR), que tiene requisitos específicos de integración de software y hardware entre el dispositivo y la plataforma. La plataforma WMR proporciona compatibilidad a nivel de sistema operativo para Lenovo Explore VR, permitiéndole funcionar de manera eficiente y ofrecer una excelente experiencia de usuario.[40], [41]

Al desarrollar juegos de RV para dispositivos de la plataforma WMR utilizando Unity, es necesario descargar el complemento "Mixed Reality Toolkit" (MRTK), diseñado específicamente para el entorno de desarrollo de Unity. Este kit de herramientas, desarrollado por Microsoft, proporciona a los desarrolladores varias funciones y componentes, como la interfaz de usuario y la gestión de entradas, lo que simplifica enormemente el proceso de desarrollo de juegos de realidad virtual para desarrolladores de juegos convencionales[42].

2.4.3 Software de modelado de código abierto Blender

Aunque los gráficos no son el enfoque principal de los juegos serios, la calidad del diseño visual en los videojuegos afecta directamente la inmersión y el disfrute del jugador. Por lo tanto, en la creación de juegos serios, mantener un nivel adecuado de diseño visual es igualmente importante para lograr los objetivos del juego [43].

En el diseño estético de este proyecto (diseño de escenarios, modelado de personajes, etc.), se eligió Blender como la herramienta principal. Blender, como un software de gráficos por computadora en 3D completamente gratuito y de código abierto, no solo ofrece una gran ventaja en términos de bajo presupuesto, sino que también es integral, ya que incluye funciones de modelado, animación y renderizado. Además, su buena compatibilidad con los motores de juego es una razón clave para elegirlo. Debido a estas ventajas, muchos desarrolladores de juegos independientes e incluso grandes estudios de juegos utilizan Blender como herramienta de modelado y renderizado de animación [44].

3 Especificaciones y restricciones de diseño

Una vez que se comprenda el desempeño de los pacientes con DPAC y se investiguen los juegos serios terapéuticos existentes, especialmente "El planeta Sonoa", se pueden establecer las especificaciones y requisitos de diseño de este proyecto combinando estos conocimientos con el marco tecnológico involucrado:

1. El juego diseñado debe ser capaz de generar efectos de audio 3D suficientemente realistas, y la mecánica del juego debe centrarse en la localización de la fuente de sonido por parte del jugador en un espacio 3D, diseñando un juego serio enfocado en el entrenamiento de la capacidad de localización del sonido.

2. Como un juego serio terapéutico cuyo propósito es ayudar a la rehabilitación de los pacientes, y dirigido a niños de 6 a 12 años, la mecánica del juego debe ser lo suficientemente divertida para asegurar la efectividad del entrenamiento, mejorar la concentración de los pacientes y motivarlos en su recuperación.

3. El juego debe tener un nivel adecuado de diseño visual y estético para aumentar la inmersión del jugador. Además, debe mantener un estilo similar al del juego principal "El planeta Sonoa", diseñando un entorno y animales con un estilo específico. Este diseño de fondo también debe ser coherente con la mecánica del juego, sin causar disonancia.

4. Como un videojuego terapéutico, debe tener suficiente adaptabilidad, es decir, debe ser capaz de ofrecer diferentes niveles de entrenamiento para pacientes con diferentes condiciones, ajustando y adaptando la dificultad según las necesidades de entrenamiento.

5. Dado que la plataforma de datos de usuario utilizada por "El planeta Sonoa" no puede aplicarse a este proyecto, para facilitar que los jugadores guarden sus configuraciones de parámetros, este proyecto debe diseñar su propio sistema de inicio de sesión de usuario.

6. Como un videojuego terapéutico, debe tener la capacidad de generar resultados de entrenamiento para que los terapeutas y los propios pacientes puedan rastrear y evaluar la situación y los efectos del entrenamiento, ajustando dinámicamente los métodos de entrenamiento y tratamiento posteriores.

7. Como un juego dirigido a niños, debe tener ciertas restricciones de contenido, evitando la aparición de cualquier contenido negativo como violencia o daño.

8. Para el diseño más detallado, como la elección de los audios utilizados para la localización y la selección del ruido de fondo, se debe buscar la opinión de profesionales para garantizar la efectividad del entrenamiento.

4 Descripción de la solución propuesta

En este capítulo se detallará el proceso de diseño y desarrollo del videojuego terapéutico "Pandalia". Primero, se discutirá el diseño conceptual del juego, describiendo el nacimiento de la idea. Luego, se explicará en detalle cada fase del desarrollo, incluyendo cómo preparar el entorno necesario para desarrollar un proyecto de RV en Unity, el diseño visual del juego, el diseño de escenarios, la gestión de usuarios, la mecánica del juego, y las funciones de ajuste de parámetros y salida de resultados. La explicación del proceso de desarrollo incluirá desde la comprensión y conceptualización del diseño hasta la implementación específica y la evaluación de los resultados del diseño.

4.1 Diseño creativo y visual del juego

4.1.1 Diseño conceptual

Durante la fase inicial de brainstorming para el diseño conceptual del juego, el primer requisito fue que el proyecto debía ajustarse al tema general de "El planeta Sonoa", es decir, el tema de los jugadores viajando en una nave espacial a un planeta compuesto por continentes con diferentes características geográficas y animales parlantes únicos en cada uno. Considerando los paisajes y animales ya presentes en "El planeta Sonoa", se descubrió que faltaban características culturales y ecológicas de Asia. La diversidad cultural podría aumentar la riqueza y el interés del juego, por lo que se decidió desde el principio que el personaje animal de este proyecto sería un panda.

Al considerar cómo combinar la mecánica del juego relacionada con la localización del sonido con pandas o la cultura asiática, se pensó en la actividad tradicional china de soltar farolillos voladores. En algunos festivales importantes, como el Año Nuevo Chino, la gente escribe deseos en farolillos de papel y luego los suelta al cielo nocturno, simbolizando la esperanza y los buenos deseos para el futuro. Por lo tanto, se decidió que el trasfondo del juego sería: los jugadores viajan a un pueblo de estilo chino habitado por pandas que están celebrando el Año Nuevo Chino. Los jugadores deben seguir las indicaciones sonoras para ayudar a los pandas a colocar los farolillos en la posición indicada por el sonido. Si la posición es correcta, el farolillo se eleva hacia el cielo. Esta idea fue discutida con los miembros del equipo, la tutora y los terapeutas, quienes reconocieron la coherencia y el atractivo de la combinación de la mecánica y el concepto.

Con el concepto definido, se nombró al proyecto y al continente habitado por pandas como "Pandalia".

4.1.2 Diseño de escenas

En el diseño del escenario, se desea crear un pequeño pueblo de estilo chino habitado por pandas. Por lo tanto, se buscaron modelos prefabricados para decorar el escenario en sitios de recursos como Unity Asset Store. Estos incluyen:

El paquete de prefabs de naturaleza forestal de baja poligonización "Free Low Poly Nature Forest" proporcionado por Pure Poly Studio [45]. El proyecto utilizó diferentes formas de montañas prefabricadas de este paquete, como se muestra en la Figura 2, para los límites del escenario del juego.

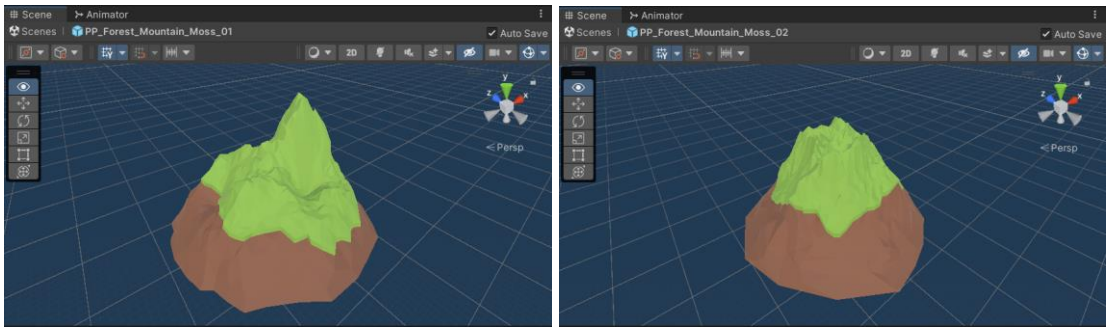


Figura 2: Los prefabs de montañas utilizados, proporcionados por Pure Poly Studio [45]**.**

También se utilizaron diferentes prefabs de pradera de este paquete, como se muestra en la Figura 3, como el terreno del juego.

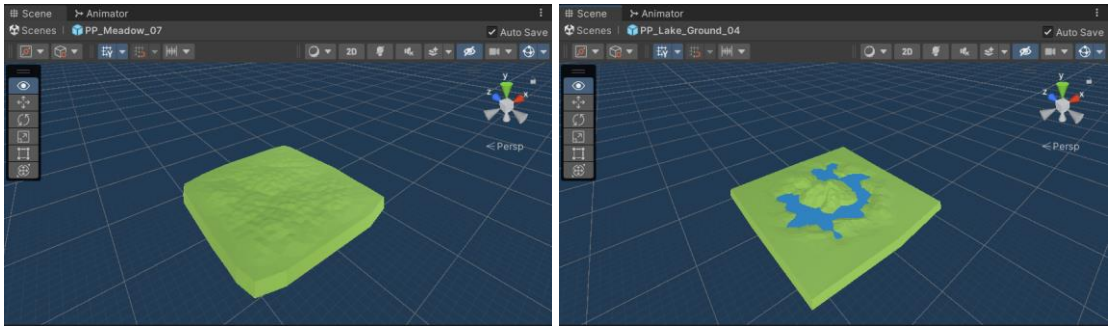


Figura 3: Los prefabs de terreno utilizados, proporcionados por Pure Poly Studio [45]**.**

Además, se usaron una serie de prefabs de flores, hierbas y árboles de este paquete como decoraciones generadas en el terreno. Cabe destacar que, debido a que los prefabs de flores y hierbas son muy pequeños, crearlos uno por uno en el escenario consumiría mucho tiempo. Por lo tanto, se desarrolló un script llamado *"VegetationSpawner.cs"* para generar aleatoriamente una cierta cantidad de objetos de vegetación dentro de un rango especificado. Para asegurarse de que solo se generen en el terreno, se utilizó RaycastHit, generando un rayo que se dispara desde el aire hacia abajo, y determinando si el objeto intersectado por el rayo tiene la etiqueta "ground". Si es así, se genera aleatoriamente uno de los prefabs de plantas según el peso de selección elegido por el jugador. La Figura 4 muestra, a la izquierda, el objeto "VegetationManager", donde se añaden diferentes prefabs de plantas en el componente del script, se configuran sus pesos y se establece la cantidad y el rango de generación. Al presionar el botón "Spawn Vegetation", se generan plantas de forma aleatoria. La derecha de la Figura 4 muestra uno de los prefabs de plantas.

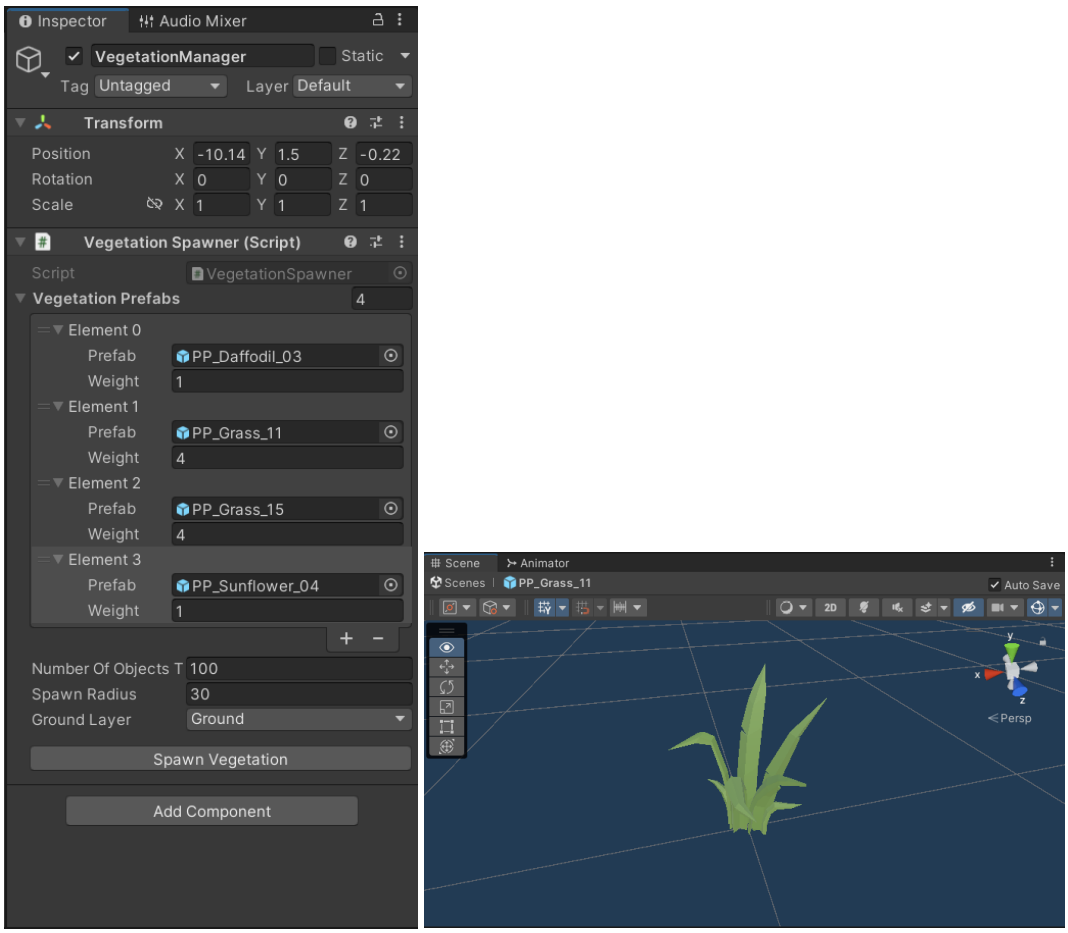


Figura 4: La componente de Script interactivo Vegetation Spawner (izquierda) y uno de los prefabs de plantas utilizado (derecha).

Por último, se utilizaron modelos de edificios de estilo oriental del paquete gratuito proporcionado por Gamemag Creation Studio [46]. Las Figuras 5 y 6 muestran los modelos de este paquete utilizados en el escenario.

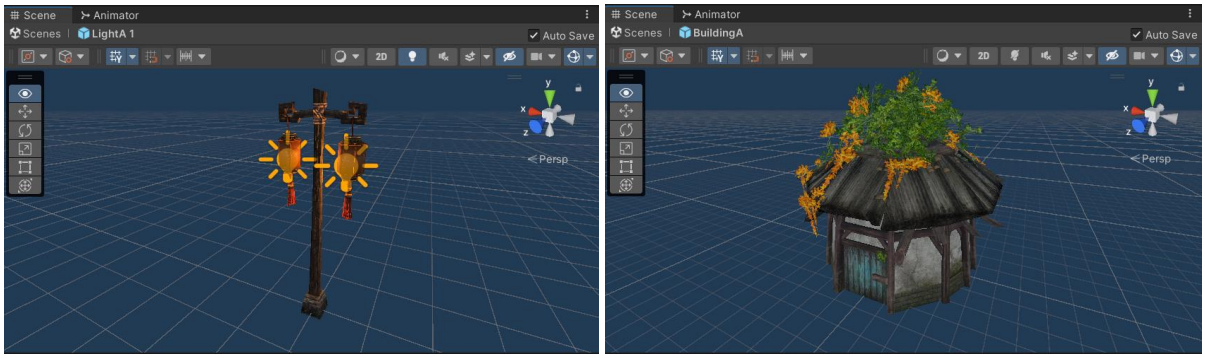


Figura 5: Los prefabs de decoración utilizados, proporcionados por Gamemag Creation Studio [46]**.**

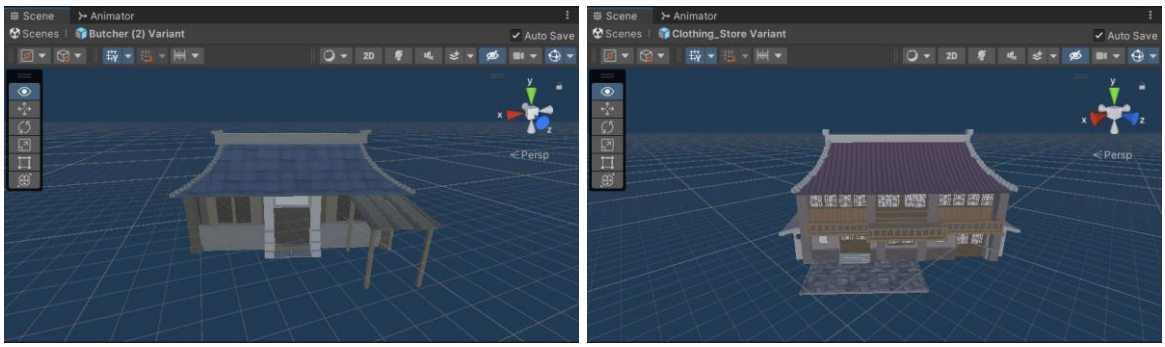


Figura 6: Los prefabs de decoración utilizados, proporcionados por Gamemag Creation Studio [46]**.**

La vista aérea del escenario final se muestra en la Figura 7, y la Figura 8 muestra algunos detalles del escenario en el juego.

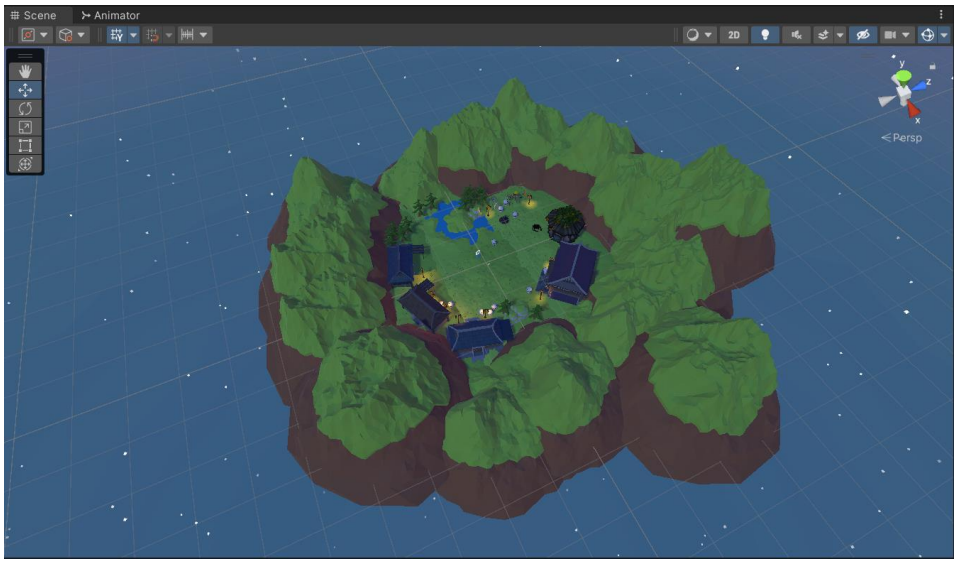


Figura 7: La vista general de la escena principal.

Figura 8: Las escenas detalladas del juego.

4.1.3 Diseño del personaje y del farolillo volador

Se diseñó una imagen simple de un panda para este proyecto utilizando Blender, y se le añadió un esqueleto para que pueda realizar movimientos de celebración cuando el jugador complete el juego, aumentando así la diversión y la sensación de logro del juego, y motivando al jugador. Las Figuras 9 y 10 muestran respectivamente la vista plana y la vista 3D del modelo en Blender.

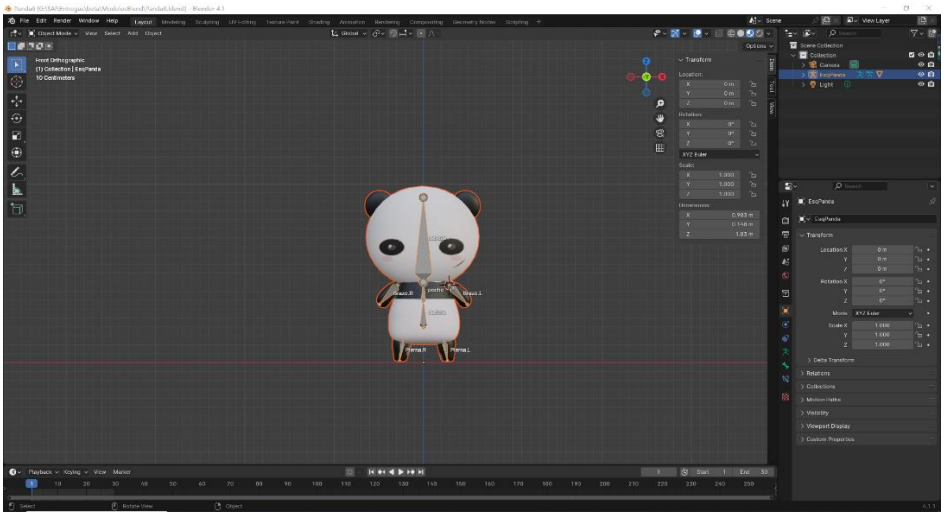


Figura 9: La vista plana del modelo de panda.

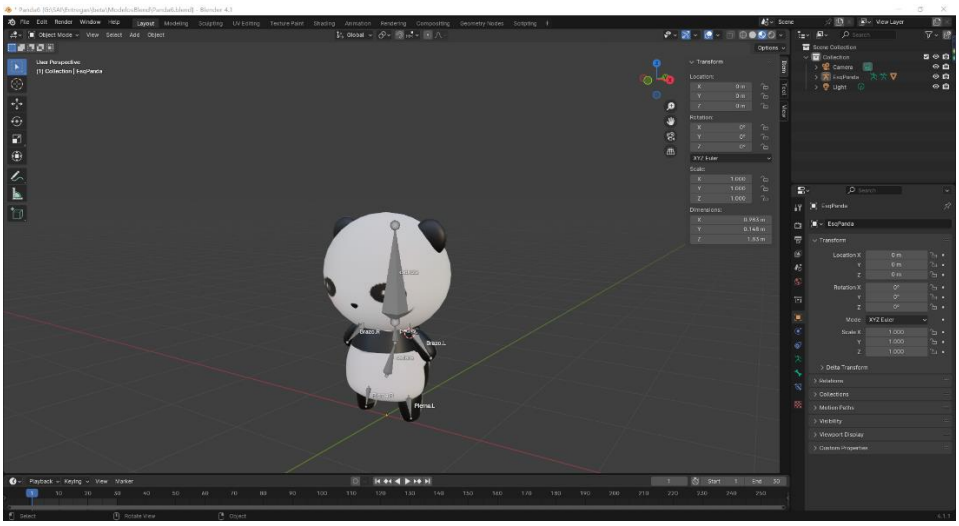


Figura 10: La vista 3D del modelo de panda.

En cuanto al diseño del personaje controlado por el jugador, dado que se trata de un juego de RV en primera persona y el jugador no puede ver al personaje durante el juego, no es necesario ningún diseño, pero se utilizan dos modelos de manos para interactuar con las farolillos generados, que son de recurso abierto y proporcionados por “Valem Tutorials”, un desarrollador independiente de videojuegos, al importarse el paquete de Unity descargado al proyecto, se obtiene los modelos como se muestra en la Figura 11 [47].

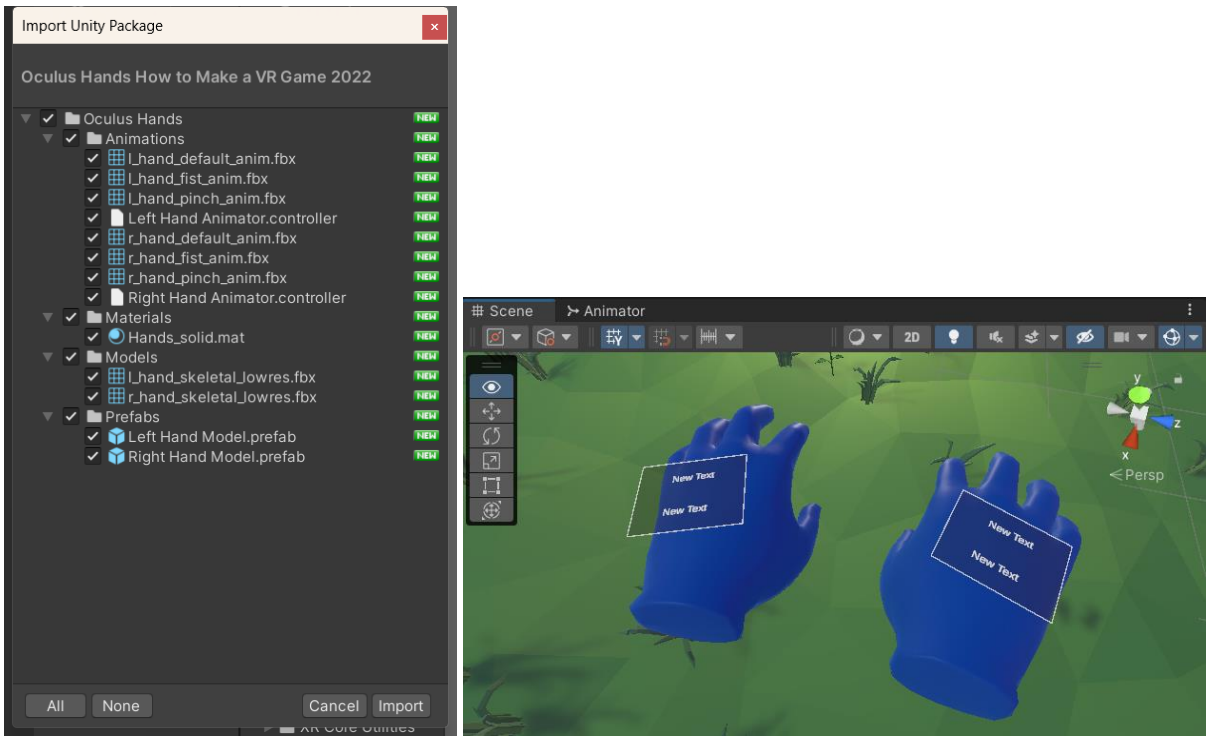


Figura 11: Los modelos de manos controlados por el jugador.

Por último, se diseñó un modelo del farolillo en Blender. Las dos capturas de arriba en la Figura 12 muestran la vista 3D en Blender del modelo, que consiste en una geometría simple para la pantalla del farolillo, con una base en la parte inferior para el encendido del farolillo volador, las dos figuras de abajo muestran el prefab del farolillo y su apariencia en el juego, respectivamente, con su material configurado como semitransparente para las características interactivas posteriores.

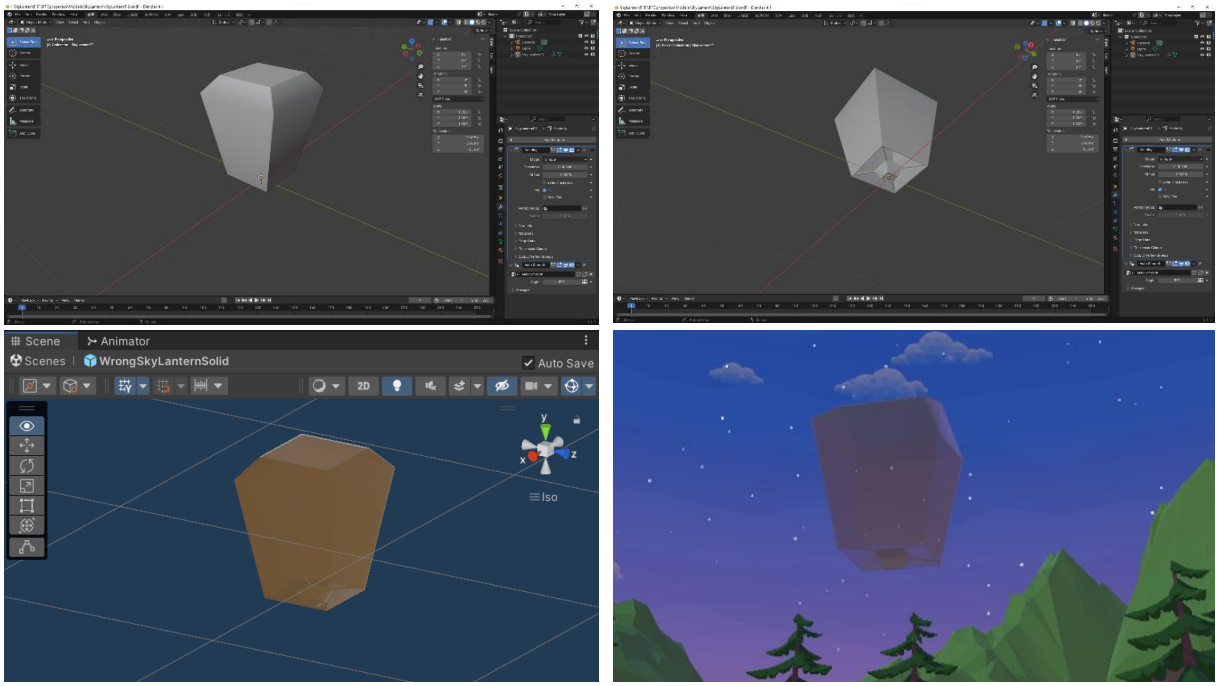


Figura 12: El modelo diseñado para el farolillo volador.

4.2 Configuración del entorno de desarrollo de juegos de RV

Unity proporciona la opción de crear proyectos de RV directamente, pero estos proyectos de RV solo ofrecen algunos paquetes y configuraciones preestablecidos. Elegir crear un proyecto 3D normal también puede permitir el desarrollo de juegos de RV.

ara el desarrollo de proyectos de RV, lo más importante es instalar XR Plug-in Management en la configuración del proyecto. Es una herramienta proporcionada por Unity para gestionar y configurar el soporte de dispositivos de realidad extendida (XR). Simplifica los pasos necesarios para configurar proyectos para diferentes plataformas XR (como AR y VR), gestionando de manera unificada los plugins y configuraciones de diferentes plataformas.

Después de instalarlo, en la misma interfaz, se debe seleccionar los Plug-in Providers correspondientes según el dispositivo de RV utilizado. En el caso de este proyecto, se elige la característica Windows Mixed Reality de OpenXR, que es el plugin adecuado para las gafas de RV Lenovo Explorer utilizadas en el proyecto. La figura 13 muestra cómo se realizó la configuración inicial del entorno RV para este proyecto.

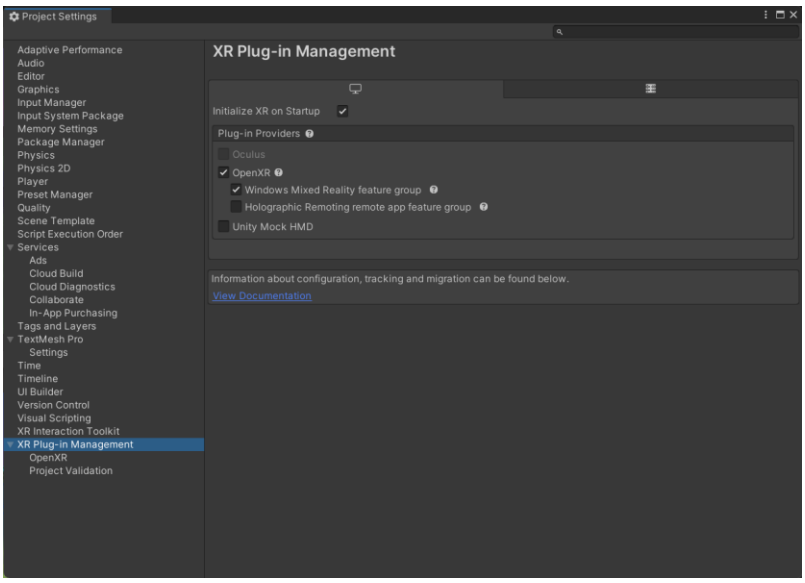


Figura 13: La ventanilla de Project settings para instalar XR Plug-in Management.

Una vez configurado, Unity permite a los usuarios agregar componentes relacionados con RV. Por ejemplo, para adaptar la cámara principal del proyecto a los efectos de RV, se debe añadir el componente “Tracked Pose Driver” (como se muestra en la figura 14) a dicha cámara, una vez añadido, recibe actualizaciones en tiempo real basadas en la posición y rotación del casco de RV conectado, y los jugadores podrán visualizar la escena del juego desde todos los lados del dispositivo de RV.

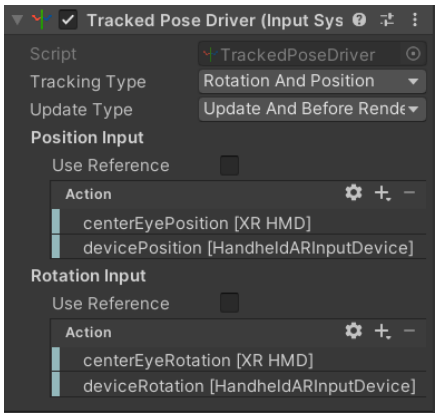


Figura 14: La componente “Tracked Pose Driver”.

Además, para configurar correctamente la función de interacción de los jugadores con los objetos del juego a través del mando de los gafas de RV, es necesario descargar el paquete “unity.xr.interaction.toolkit”. A través de una serie de configuraciones, el sistema puede leer las entradas del mando RV. Los detalles específicos de la configuración se explicarán en el ANEXO A.

4.3 Estructura del juego

Una vez explicada la jugabilidad y mecánica básica del juego, y comprendida la configuración básica del entorno del proyecto de RV, a continuación, se describe en profundidad el método de implementación de la mecánica del juego. Este capítulo explicará la estructura del juego siguiendo las etapas principales del flujo del juego. En los siguientes capítulos, se detallará cómo se implementa cada etapa, centrándose en cómo se desarrollan las diferentes funciones a través de scripts de programación.

Primero, al iniciar el juego, no se entra inmediatamente en el entorno de RV, sino que se accede a la etapa de inicio de sesión del usuario. Los usuarios deben realizar la operación de inicio de sesión utilizando el teclado y el ratón en la computadora. En esta etapa, el sistema permite a los jugadores crear nuevos usuarios o iniciar sesión con una cuenta existente. El sistema creará un archivo .JSON para el nuevo usuario según un modelo de datos preestablecido, en el cual se guardarán todas las configuraciones relacionadas con el juego del usuario para que pueda leerlas directamente en futuros inicios de sesión.

Después de iniciar sesión o crear un nuevo usuario con éxito, el usuario entra inmediatamente en la interfaz de configuración del juego. En esta etapa, el terapeuta o el jugador pueden ajustar la dificultad del juego y la configuración de volumen según la situación. Se puede decidir si los cambios de configuración se aplicarán solo para la sesión actual o se guardarán en los datos del usuario. El sistema permite personalizar la dificultad para hasta cinco niveles (del 1 al 5). En cada nivel, hay varios parámetros que el usuario puede ajustar libremente. En cada sesión de juego, el usuario puede decidir jugar solo un nivel de dificultad o varios (hasta cinco).

Después de ajustar la configuración, se entra en la fase principal del juego. En esta etapa, el juego entra en modo RV. El jugador debe ponerse el dispositivo de RV y los controladores para comenzar a jugar. Primero, el jugador entra en la escena principal del juego, el pueblo de los pandas. En este momento, el entrenamiento no comienza de inmediato, sino que el jugador puede pasear por el pueblo y explorar libremente la escena. En el escenario, los pandas explicarán al jugador la tradición de lanzar farolillos voladores y lo guiarán para comenzar el juego. Alternativamente, el jugador puede caminar directamente hacia el pilar de luz translúcido en la escena. Entrar en el pilar de luz significa que comienza el juego de localización de sonido. Después de comenzar, los movimientos del jugador se limitarán, y hasta que complete el juego o presione el botón de finalización en el controlador de RV, el jugador solo podrá cambiar la perspectiva girando la cabeza.

En este momento, dentro de un cierto rango alrededor del jugador, se generará una cantidad aleatoria de farolillos voladores en posiciones aleatorias. Estos farolillos tienen la misma apariencia, pero solo uno de ellos emitirá un sonido. El jugador debe localizar el sonido y seleccionar el objeto fuente utilizando el controlador (en el juego se verá un rayo que se usa para seleccionar objetos). Si selecciona incorrectamente, el farolillo equivocado será destruido. Si selecciona correctamente, encenderá el farolillo correcto, haciéndolo elevarse.

Seleccionar bien una vez cuenta como completar una ronda del juego. Después de seleccionar correctamente el farolillo sonoro, comienza la siguiente ronda inmediatamente hasta completar el número de rondas establecidas para ese nivel de dificultad. Al completar todas las rondas de un nivel, si hay un nivel de dificultad siguiente, el juego avanza al siguiente nivel.

Si el jugador completa todos los niveles de dificultad o presiona el botón de salida, el juego terminará y se le preguntará si desea exportar los resultados del juego actual. Luego, el jugador puede elegir volver a la pantalla de configuración para ajustar la configuración y jugar nuevamente, o salir del juego directamente. La figura a continuación muestra el diagrama de flujo de un juego completo.

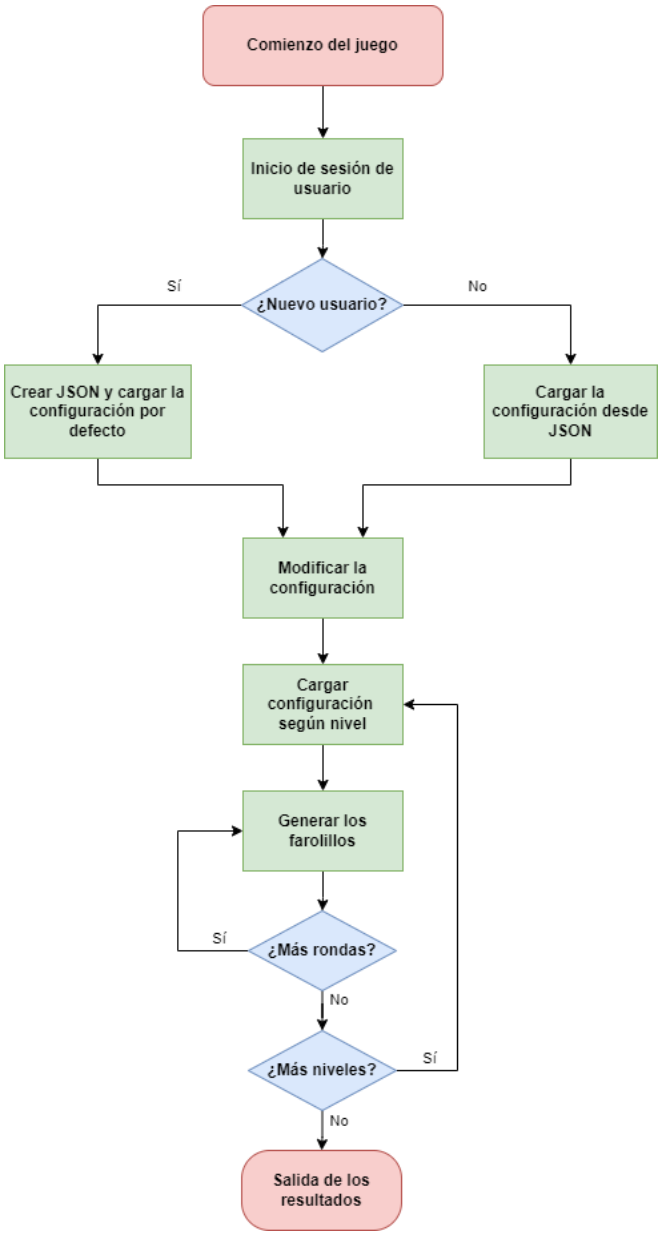


Figura 15: Diagrama de flujo de un juego completo.

4.4 Mecanismos de acceso y gestión de usuarios

Después de resumir el flujo estructural del juego, se procederá a explicar en detalle la implementación de cada parte importante del flujo del juego. Primero, en este capítulo se presentará la importancia de la gestión de usuarios en el proyecto y el método de implementación de su mecanismo. Esta función desempeña un papel central en un juego de naturaleza terapéutica, no solo garantizando la personalización de la configuración del juego, sino también asegurando la seguridad y la protección de la privacidad de los datos del usuario.

La arquitectura del sistema de gestión de usuarios adopta una estrategia modular, compuesta principalmente por tres componentes clave: *"User.cs"*, *"LoginManager.cs"* y *"SettingsManager.cs"*. Cada componente asume diferentes responsabilidades e interactúan entre sí, formando conjuntamente un sistema de gestión de usuarios eficiente y seguro.

- **"User.cs"**: Es una clase de modelo de datos que define y almacena la información básica del usuario, como nombre de usuario, contraseña y varias configuraciones personalizadas del juego (como control de volumen, selección de dificultad, etc.). El diseño de esta clase está directamente relacionado con la personalización de la experiencia del usuario y la flexibilidad de la configuración del juego.

- **"LoginManager.cs"**: Es responsable de manejar la lógica de inicio de sesión del usuario. Desde la verificación de la identidad del usuario hasta la carga de datos del usuario después del inicio de sesión, este componente garantiza que el usuario pueda acceder al juego de manera segura y fluida. A través de la interacción con la clase *"User.cs",* verifica las credenciales proporcionadas por el usuario y permite o niega el acceso en consecuencia.

- **"SettingsManager.cs"**: Gestiona las opciones de configuración del usuario en el juego, incluyendo la guarda y carga de configuraciones personalizadas del usuario. Este componente permite que las configuraciones del entorno de juego del usuario se mantengan consistentes en diferentes sesiones de juego al leer y escribir el archivo de configuración del usuario.

4.4.1 "User.cs"

La clase User define todos los parámetros personalizables del jugador en el juego. La construcción de esta clase es la base que determina el grado de adaptabilidad del juego. Los datos del usuario se guardan según la estructura de esta clase. Comprender la estructura y los atributos específicos de esta clase es crucial para entender el diseño posterior del proyecto. La clase User contiene las siguientes tres categorías de atributos:

Datos Básicos: Cada usuario tiene un username y un password, que se utilizan para gestionar el acceso y la identificación en el sistema.

Configuraciones de Volumen: Se manejan múltiples ajustes de volumen para personalizar la experiencia auditiva del usuario. Esto incluye:

- **masterVolume**: Volumen maestro, controla el volumen global del juego.
- **bgmVolume**: Volumen del ruido del fondo.
- **feedbackVolume**: Volumen de los efectos sonoros de retroalimentación, es decir, el sonido que se reproduce cuando el jugador completa una ronda con éxito o fracasa en el intento.
- **objectsVolume**: Volumen de los farolillos voladores sonoros en el entorno del juego.

Configuraciones de Terapia: En la clase de usuario también define una clase *"TherapySettings"*, los atributos definidos en esta clase determinan todos los ajustes que se pueden personalizar en el entrenamiento, que son específicos para cada nivel de dificultad del juego, y el sistema creará uno de esta estructura para cada nivel de dificultad para cada usuario. Estos ajustes se inicializan con valores por defecto y pueden personalizarse en función de los progresos y necesidades del usuario. En dicha clase, para cada nivel de dificultad, se definen las siguientes propiedades:

- **numObjects**: Número de farolillos voladores que se generan en una ronda de juego, de los cuales sólo se suena uno.
- **Distance**: Distancia máxima que puede generar un farolillo.
- **height**: Altura máxima que puede generar un farolillo.
- **timeLimit**: Límite de tiempo para identificar el farolillo sonoro en una ronda de juego.
- **roundsToPlay**: Número de rondas a jugar en cada sesión de nivel de dificultad.
- **audioInterval**: Intervalo de tiempo entre sonidos emitidos por los farolillos.
- **selectedAudioClips**: Clips de audio seleccionados para reproducirse en ese nivel.

Además, la clase *"User"* define un único método llamado *"SetDefaultValuesForLevel"*, el cual solo se invoca una vez al crear un nuevo usuario. Este método establece un conjunto predeterminado de *"TherapySettings"* para cada nivel de dificultad del nuevo usuario y escribe estas configuraciones en el archivo JSON del nuevo usuario. Según aumenta el nivel de dificultad, se ajustan propiedades como la cantidad de objetos, la distancia y el intervalo de audio, proporcionando así un proceso adecuado y desafiante para el usuario.

4.4.2 "LoginManager.cs"

Esta clase es responsable de toda la lógica sobre la gestión de login de usuarios a través de una serie de métodos, su implementación específica es la siguiente:

Método y Flujo de Autenticación：

1. Inicio de Sesión:

Verificación de Campos Vacíos: Este método primero verifica que los campos de nombre de usuario y contraseña no estén vacíos utilizando condiciones básicas if. Si alguno de los campos está vacío, se muestra un diálogo informando al usuario que ambos campos son necesarios.

Llamada al Método de Manejo de Login (HandleLogin Método): Si los campos están completos, OnLoginButtonClicked llama al método HandleLogin, pasando el nombre de usuario y la contraseña ingresados.

A continuación, se presentan dos conjuntos de imágenes que muestran varias situaciones durante el inicio de sesión del usuario. En la figura 16, se muestra que cuando los campos de nombre de usuario y contraseña están vacíos, el sistema muestra un error y el usuario debe cerrar el cuadro de diálogo y volver a ingresar los datos. En la figura 17, se muestra la situación en la que se ha introducido un nuevo nombre de usuario. En la figura superior derecha, el sistema pregunta al usuario si desea crear un nuevo usuario con ese nombre. La figura inferior izquierda muestra el resultado de intentar iniciar sesión con un nombre de usuario existente y una contraseña incorrecta. La última figura muestra que, ya sea que se acepte crear un nuevo usuario o que se inicie sesión correctamente con un nombre de usuario y contraseña correctos, el sistema indica que el inicio de sesión fue exitoso y se accede directamente a la pantalla de configuración.

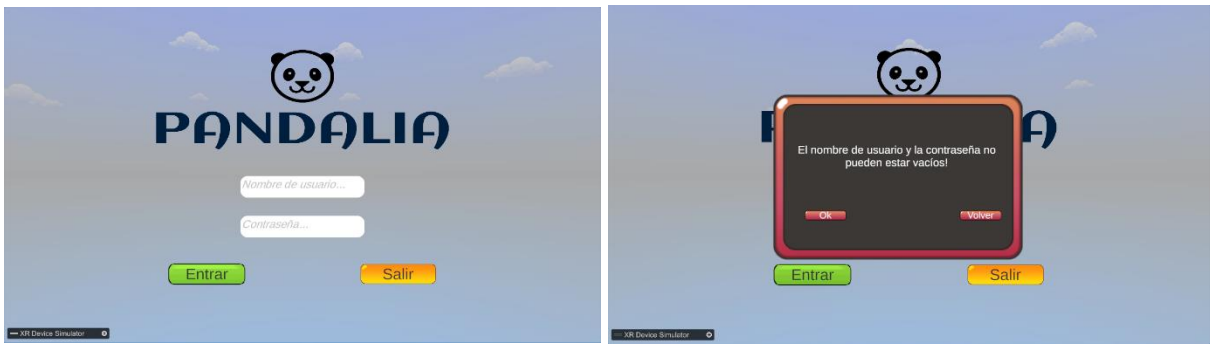


Figura 16: Pantalla de inicio de sesión del juego (izquierda), cuadro de dialogo que muestra información relevante.

Figura 17: Los diferentes casos de inicio de sesión de usuario.

2. Manejo del Login (“HandleLogin” Método):

Verificación de Existencia del Archivo de Usuario: Después de recibir el nombre de usuario y la contraseña ingresados por el jugador, el método *“HandleLogin”* primero llamará al método *“E*ncrypt” de la clase *“SimpleEncryption”* para convertir el nombre de usuario de un dato tipo string ordinario a una cadena codificada en Base64. La razón para hacer esto es que, al crear un usuario, el sistema utilizará este método para encriptar el nombre de usuario y la contraseña, y usará el nombre de usuario encriptado como el nombre del archivo JSON que almacena los datos de ese usuario. Dado que el sistema determina si un usuario ya existe comparando si existe un archivo JSON con el nombre del usuario encriptado en el almacenamiento local, en este paso primero se encripta el nombre de usuario ingresado utilizando el mismo método.

Validación de la Contraseña: Si el nombre de usuario no existe, se preguntará al usuario si desea crear un nuevo usuario. En caso de confirmar que el nombre de usuario existe, el sistema usará *“SimpleEncryption.Decrypt”* para desencriptar la contraseña encontrada en el archivo JSON del usuario local y la comparará con la contraseña ingresada. Si coinciden, el sistema mostrará un mensaje de inicio de sesión exitoso y accederá a la pantalla de configuración, cargando las configuraciones del archivo JSON del usuario.

Es importante destacar que, una vez que el usuario ha iniciado sesión correctamente, el método *“HandleLogin”* deserializa los datos del archivo JSON del usuario, convirtiendo la cadena de texto en formato JSON a un objeto en C#. Primero, se guardan todas las cadenas de datos JSON en una variable string, y luego, utilizando la clase JsonUtility de Unity, que maneja la serialización y deserialización de datos JSON, se convierte toda la información del archivo local del usuario a la estructura de la clase *“User”*, cargándola en una instancia de la clase *“User”* llamada *“loadedUser”*. Con esta instancia, se puede pasar a cualquier script o componente que necesite interactuar con los datos del usuario, como *“SettingsManager.cs”.*

Además, después de obtener el objeto usuario “*loadedUser*” construido a partir de los datos JSON, es necesario realizar una operación para cargar los datos de *loadedUser* en la interfaz de usuario (UI) del juego. Dado que, al iniciar el juego, antes de que el usuario inicie sesión, el sistema no puede asignar valores correspondientes a todos los componentes de la UI, solo después de que el usuario haya iniciado sesión correctamente es posible realizar esta operación. Por lo tanto, después del inicio de sesión del usuario, el sistema debe utilizar algún método para notificar a los scripts relacionados con la interacción de la UI para que carguen los valores de la UI. El mecanismo de eventos en C# puede realizar esta tarea a la perfección. Este mecanismo permite que una clase u objeto notifique a otras clases u objetos que ha ocurrido algo. Basado en el patrón de publicación-suscripción implementado mediante delegados, permite que el publicador (en este caso, “*LoginManager.cs*”) notifique a uno o más suscriptores (en este caso, las clases que necesitan interactuar con la UI, como “*TherapySettingsManager*.*cs*”) de un evento sin conocer su implementación específica.

En la clase "*LoginManager*", se usa el tipo de delegado Action para definir un evento "*OnUserLoggedIn*". Una vez que el sistema verifica que la información de inicio de sesión del usuario es correcta, llamará a "*OnUserLoggedIn?.Invoke()*" para disparar el evento "*OnUserLoggedIn*". En otras clases, se pueden suscribir métodos que necesitan ejecutarse después del inicio de sesión al evento, por ejemplo, en TherapySettingsManager.cs se puede suscribir el método "*LoadTherapySettings*" al evento "*OnUserLoggedIn*". Así, después del inicio de sesión, esta clase recibirá la notificación y ejecutará el método "*LoadTherapySettings*" para cargar las configuraciones del juego desde "*loadedUser*" a los componentes de la UI.

La ventaja de este método es que "*LoginManager*" no necesita conocer qué operaciones específicas se ejecutarán después del inicio de sesión del usuario. Solo se encarga de publicar el evento en el momento adecuado, sin necesidad de crear una instancia de la clase "*TherapySettingsManager*" para llamar a sus métodos, lo que reduce el acoplamiento entre el código y mejora la mantenibilidad y extensibilidad de este.

3. Creación de Usuario ("CreateUser" Método):

Como se mencionó anteriormente, cuando el nombre de usuario no existe en los archivos del sistema, el usuario puede optar por crear ese usuario. El proceso específico es el siguiente:

Encriptación de Credenciales:

Una vez que el usuario confirma la creación del usuario, el sistema llamará inmediatamente al método "*Encrypt*" de "*SimpleEncryption*" para encriptar el nombre de usuario y la contraseña ingresados. Este método convierte las cadenas de texto encriptadas en cadenas codificadas en Base64 para aumentar la seguridad durante el almacenamiento y la transmisión. Primero, convierte la cadena en datos binarios aptos para operaciones de encriptación y luego convierte el arreglo de bytes en una cadena codificada en Base64. Base64 es un método de codificación común que convierte datos binarios en formato de texto, facilitando la transmisión en sistemas que no admiten datos binarios. Aunque este método es relativamente simple y puede ser fácilmente revertido a la cadena de texto original mediante el método "*Convert.FromBase64String*" de C#, ofrece seguridad relativa en el almacenamiento y la transmisión, dificultando su lectura y extracción en el sistema local, aunque no proporciona una verdadera confidencialidad.

Inicialización de Configuraciones del Usuario:

Después de completar la encriptación del nombre de usuario y la contraseña, el sistema crea un nuevo objeto User. Al crearse, el objeto User llama automáticamente al método "*SetDefaultValuesForLevel*" para establecer valores predeterminados en la configuración del juego. Luego, el sistema utiliza el nombre de usuario encriptado como el nombre para crear un archivo JSON correspondiente al usuario y almacena en este archivo el nombre de usuario y la contraseña encriptados, junto con la configuración inicial del juego.

Notificación y Cambio de Pantalla:

En este flujo, otra función muy importante es la de mostrar cuadros de diálogo con mensajes pertinentes para el usuario, proporcionando la orientación necesaria en casos como fallo de inicio de sesión, éxito de inicio de sesión o registro de un nuevo usuario. Esta función es manejada principalmente por el método "*ShowConfirmDialog*". Este método acepta un parámetro de tipo string para especificar el texto del mensaje que se mostrará en el cuadro de diálogo, un parámetro de tipo delegado Action que se invoca cuando el usuario hace clic en el botón "Confirmar" y otro parámetro de tipo delegado Action que se invoca cuando el usuario hace clic en el botón "Cancelar". Además, es necesario añadir escuchadores a estos dos botones para que el sistema pueda monitorear en tiempo real cuándo se presionan.

Este método proporciona una forma flexible de mostrar cuadros de diálogo con operaciones de confirmación y cancelación, permitiendo a los llamadores definir el comportamiento específico de estas operaciones mediante delegados. Este enfoque permite que el comportamiento del cuadro de diálogo se personalice dinámicamente según el contexto, mejorando la reutilización y la flexibilidad del código.

Finalmente, después de un inicio de sesión o registro exitoso, el sistema llama a "*SwitchToSettings*" para cambiar de la página de inicio de sesión a la página de configuración.

4.4.3 SettingsManager.cs

El "SettingsManager" es una clase crucial dentro del proyecto, encargada de manejar la persistencia y recuperación de las configuraciones de usuario. Este componente asegura que las preferencias individuales de los usuarios se mantengan consistentes a través de las sesiones del juego, ofreciendo una experiencia personalizada y coherente. A continuación, se detalla su implementación y funcionalidad principal.

Persistencia de las configuraciones： Para permitir que el sistema transfiera configuraciones locales al escenario del juego, hay dos puntos clave. Primero, dentro de "*SettingsManager*", se define una instancia de la clase "*User*" llamada CurrentUser para recibir el usuario cargado deserializado por "*LoginManager*" durante el inicio de sesión. En segundo lugar, es necesario implementar "*SettingsManager*" como un patrón singleton, que generalmente se usa en escenarios donde se necesita controlar el acceso a los recursos o compartir un estado global en todo el proyecto, como en el caso de administradores de juegos, de audio o de configuración. Al configurar "*SettingsManager*" como un singleton, este mantiene la instancia de CurrentUser que guarda todas las configuraciones, y acompaña las acciones del jugador, cargándose desde el escenario de configuración al escenario de juego.

Método para guardar la nueva configuración:

Además, en "*SettingsManager*" se define un método crucial llamado "*SaveCurrentSettings*", utilizado para guardar en la interfaz de configuración las nuevas configuraciones modificadas por el usuario. Este método crea un archivo JSON en el directorio de datos persistentes de la

aplicación utilizando el nombre de usuario, para establecer la ruta del archivo donde se guardan las configuraciones. Luego, serializa el objeto de usuario que contiene las configuraciones actuales en formato JSON y escribe este JSON en el archivo especificado. Este método es el único en el sistema que puede modificar las configuraciones del usuario, asegurando así la seguridad de los datos del usuario.

4.4.4 Interacción entre Componentes

Finalmente, una breve descripción de la interacción entre estos tres componentes: Cuando un usuario inicia sesión a través de "*LoginManager*", se verifica su identidad. Si las credenciales son correctas, "*LoginManager*" carga el perfil del usuario desde el almacenamiento local y notifica a "*SettingsManager*" para guardar esa instancia de usuario. Luego, el administrador de configuraciones actualiza las configuraciones del sistema para asegurar que el ambiente del juego esté adecuadamente personalizado. Si se necesita crear un nuevo usuario, "*LoginManager*" recoge la información necesaria y pide a "*SettingsManager*" que use configuraciones predeterminadas para inicializar un nuevo perfil, que luego se personaliza según las interacciones iniciales del usuario con el juego. Durante o después de la sesión de juego, cualquier cambio en las configuraciones del usuario es administrado por "*SettingsManager*" para asegurar que todas las modificaciones se guarden efectivamente y estén disponibles para la próxima sesión del usuario. En la figura 18 a continuación, se muestra el diagrama de flujo de trabajo de estas funciones durante esta fase.

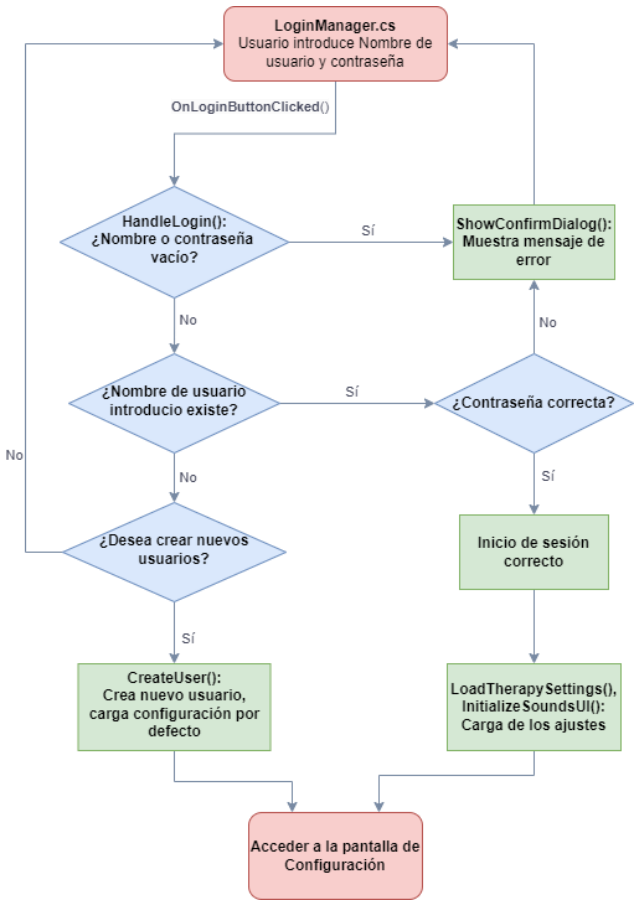


Figura 18: Diagrama de flujo de la mecánica la gestión de usuario.

4.5 Mecanismo de los ajustes personalizados para uso terapéutico y su diseño de UI correspondientes

Tras explicar la implementación de la gestión de usuarios, ahora procederemos a detallar la segunda fase del juego: la fase de configuración personalizada. Esta parte se implementa mediante un diseño integrado que combina los métodos de modificación de configuración con los métodos de interacción de la interfaz de usuario (UI) en un solo script. El objetivo de este diseño es asegurar que la interfaz de usuario (UI) y las configuraciones funcionales del juego estén estrechamente integradas, proporcionando así una experiencia de usuario sin fisuras. Este diseño integrado permite a los usuarios ajustar configuraciones de manera intuitiva y ver inmediatamente los efectos de estos cambios en el juego.

En esta sección se han desarrollado dos componentes principales:

- **"SoundSettingsManager"**: Encargado de toda la lógica de modificación de las configuraciones de sonido y su interacción con los componentes de la UI correspondiente. Además, hereda varios métodos importantes relacionados con la reproducción de audio.
- **"TherapySettingsManager.cs"**: Responsable de todas las configuraciones personalizadas relacionadas con el contenido del juego de entrenamiento y su interacción con los componentes de la UI.

4.5.1 Ajustes de los sonidos ("SoundSettingsManager.cs")

Inicialización y gestión de instancias: Primero, este método implementa el patrón singleton para asegurar la existencia de una única instancia de "*SoundSettingsManager*", que integra fuentes de audio del juego como el ruido de fondo, crucial para mantener la coherencia en la configuración del sonido en diferentes partes del juego. A continuación, en el método "*OnEnable*" de la clase, se subscribe al evento "*OnUserLoggedIn*" de la clase "*LoginMaster*" mediante el método "*LoadVolumesSettings*". Esto permite que el sistema active inmediatamente el método "*LoadVolumesSettings*" tras el inicio de sesión del usuario, asignando los valores relacionados con el volumen de CurrentUser, instanciado en el singleton de "*SettingsManager*", a los componentes de la UI, incluyendo varios sliders.

Implementación del control de volumen: En Unity, AudioMixer es un componente clave para gestionar la salida de audio, permitiendo a los desarrolladores controlar precisamente el volumen de diversas fuentes de sonido. En este proyecto, hay tres tipos diferentes de sonidos: ruido de fondo que aumenta la dificultad del juego, sonidos emitidos por el farolillo volador sonoro que los jugadores necesitan localizar, y sonidos de notificación que el sistema emite cuando el jugador completa o no completa el juego. Además, se ha añadido un volumen principal para aumentar o disminuir todos los tipos de efectos de sonido simultáneamente. Usando AudioMixer, es posible controlar individualmente estos diferentes efectos de sonido, ofreciendo una experiencia auditiva altamente personalizada.

- **Configuración del AudioMixer**: Mediante el AudioMixer de Unity, se pueden crear múltiples canales de audio, cada uno controlando una fuente de sonido diferente (como máster, ruido, etc.). En el script de "*SoundSettingsManager*", cada cambio en el valor de los sliders de volumen llama a un método "*SetVolume"*, que primero usa el método "*ConvertSliderValueToVolume*" para convertir el valor del slider a decibelios (dB), luego llama al método "*AudioMixer.SetFloat*", pasando un string (que representa el nombre del canal de audio) y un número flotante (que representa el nivel de volumen), y finalmente asigna el valor de volumen de nuevo a CurrentUser. Este método de "*SetVolume*" se añade como listener a su respectivo slider de la UI, asegurando que el slider pueda leer los valores de la base de datos y permitiendo ajustes de volumen en tiempo real que se resignan a CurrentUser.

- **Método "ConvertSliderValueToVolume"**: Este método convierte el valor lineal de un slider de la UI en un valor de decibelios no lineal para el control de volumen. Como la percepción del sonido por el oído humano es logarítmica, usar directamente el valor lineal del slider como volumen puede hacer que los ajustes parezcan insuficientes. El método utiliza funciones como "*Mathf.Lerp*" y "*Mathf.Sqrt*" para mapear el valor lineal del slider al rango logarítmico de decibelios, haciendo que los incrementos y decrementos de volumen sean más naturales y coincidan con la percepción auditiva humana.

Implementación detallada de la carga dinámica de sonido: Este proyecto ofrece una variedad de audio para diferentes necesidades, incluyendo actualmente seis tipos diferentes de ruidos de fondo y más de diez cortos clips de audio para que los usuarios localicen. Considerando la gran cantidad de clips de audio, si se agregaran manualmente al proyecto, incrementaría el trabajo innecesariamente y complicaría la adición de más recursos en el futuro. La carga dinámica de sonido es un componente crucial del diseño de juegos, especialmente cuando se necesita cambiar rápidamente los recursos de audio según la selección del usuario o cambios en el escenario del juego.

- **Método "LoadAudioClips"**: Utilizando el método "*Resources.LoadAll*", se pueden cargar archivos de audio en tiempo de ejecución desde una carpeta de recursos predefinida. Este método permite empaquetar los archivos de audio como recursos en el juego, en lugar de codificarlos duramente en los scripts del juego. Por ejemplo, cargar todos los archivos de tipo AudioClip desde la carpeta "Audio/Clips". Estos archivos de audio luego se agregan a las listas correspondientes (como bgmSounds, lanternSounds) para su uso en otras funciones del juego.

- **Ventajas**: La mayor ventaja de la carga dinámica es su flexibilidad y escalabilidad. Los desarrolladores pueden fácilmente agregar o reemplazar recursos de audio sin modificar o recompilar el código. Esto es particularmente útil para aplicaciones que necesitan actualizar frecuentemente el contenido de audio, como cambiar la música de fondo durante diferentes festividades o eventos.

- **Integración con la interacción del usuario**: Cuando los usuarios cambian la configuración de audio a través de la UI (como seleccionar diferentes músicas de fondo), "*SoundSettingsManager*" carga y reproduce de inmediato el nuevo audio seleccionado. Esto proporciona retroalimentación instantánea, aumentando la inmersión y satisfacción del usuario.

En la figura 19 a la izquierda, se muestra todos los archivos de ruido de fondo actualmente almacenados en la carpeta Assets\Resources\Audio\Noises. Después de llamar al método "*LoadAudioClips*", el sistema carga automáticamente todos los archivos de audio de esa carpeta y los agrega al componente desplegable de la UI mostrado en la figura 19 a la derecha.

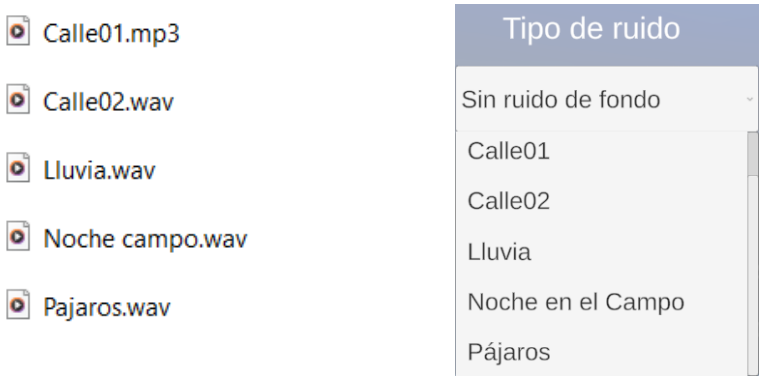


Figura 19: Los archivos de audio de ruido de fondo en la carpeta "resources" (izquierda), y ruidos de fondo cargados en UI del juego (derecha).

Diseño de UI: En la figura 20 que se describe, la interfaz de usuario (UI) para la configuración de volumen presenta cuatro Sliders de colores que controlan el volumen de cuatro tipos diferentes de sonidos en el juego. Deslizando los Sliders de izquierda a derecha, se puede aumentar el volumen, y al lado derecho de los Sliders se muestra el valor de volumen en decibelios (dB). Se observa que el máximo volumen, con un valor logarítmico de 0 dB en el **AudioMixer** de Unity, representa el volumen máximo que el canal puede emitir sin distorsión. Importante destacar que 0 dB no significa "silencio", sino el máximo volumen alcanzable sin incrementar la distorsión. Cuando el volumen se configura por debajo de 0 dB, los valores negativos indican una reducción respecto al volumen máximo. Por ejemplo, -10 dB representa aproximadamente un tercio de la energía sonora del máximo volumen. En la escala logarítmica de decibelios, -80 dB, aunque técnicamente no es completamente silencioso, es un nivel de sonido muy bajo y casi inaudible para la mayoría de las personas. Establecer -80 dB como el valor mínimo de volumen permite efectivamente un silencio completo, manteniendo la flexibilidad en el control. El rango de 0 dB a -80 dB permite un ajuste detallado del volumen, lo cual es extremadamente útil para manejar la música de fondo, los sonidos ambientales y los efectos sonoros, especialmente cuando se requiere un control preciso de las capas y profundidad de sonido.

Además, se han implementado dos botones específicos para el volumen del farolillo sonoro y el volumen de los efectos de sonido. Dado que no se puede escuchar el ajuste de estos

volúmenes en tiempo real como ocurre con el ruido de fondo, es necesario implementar métodos específicos, como "*TryObjVolume*" y "*TrySfxVolume*", para probar estos volúmenes.

Se utiliza un componente de tipo drop down para cambiar el ruido de fondo, lo cual tiene la ventaja de que, al seleccionar un audio diferente en la lista desplegable, se puede reproducir inmediatamente el nuevo ruido de fondo sin ocupar espacio adicional en la interfaz de usuario.

Finalmente, en la parte inferior de la figura, el botón Guardar activa el método "*SaveCurrentSettings*" de "*SettingsManager*". Si no se presiona este botón, cualquier ajuste modificado solo será efectivo durante la sesión actual del juego. Solo después de modificar los ajustes y presionar Guardar, los cambios se guardarán en el archivo JSON del usuario, asegurando que las preferencias se mantengan para futuras sesiones.

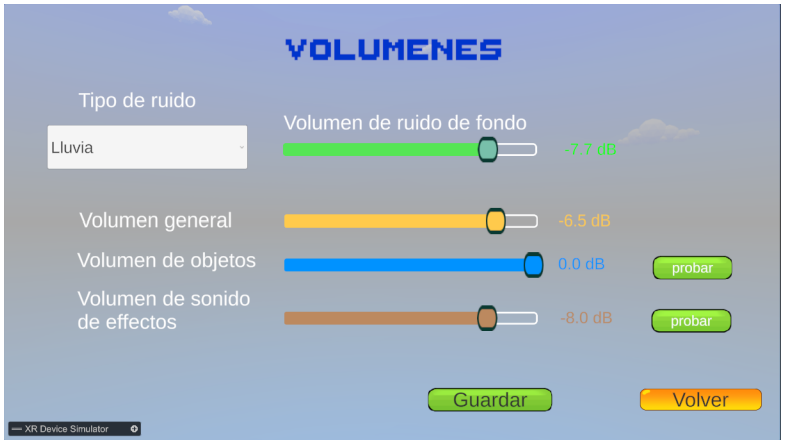


Figura 20: UI de la configuración de volúmenes del juego.

4.5.2 Ajustes de la dificultad del juego ("TherapySettingsManager.cs")

El "*TherapySettingsManager*" no se implementa como un singleton y solo existe en la fase de configuración. Una vez que el jugador entra en el juego, este componente se destruye. Esto se hace para que las configuraciones del juego no se modifiquen por acciones humanas o del sistema durante el juego. Las configuraciones se leen únicamente a través del CurrentUser en "*SettingsManager*", garantizando así la estabilidad del sistema y la seguridad de los datos.

Parámetros relacionados con la dificultad: La figura 21 muestra la interfaz de configuración para el modo de entrenamiento terapéutico del juego. En esta sección, se explicarán detalladamente los distintos ajustes ajustables y cómo estos implementan la flexibilidad en la dificultad del juego, además de cómo se realizan estas configuraciones en interacción con la UI.

Figura 21: UI de configuración del modo de terapia del juego.

- **Numero de farolillos a generar**: Define la cantidad de farolillos que se generan en cada ronda de juego. Este número se ajusta mediante el método "*SetNumObj*", que está vinculado a un componente de Input Field en la UI. El usuario establece este valor ingresando un número en el componente. El método "*SetNumObj*" limita los valores ingresados entre 2 y 10, convirtiendo cualquier otro número o caracteres en un 3. El motivo de establecer 10 como máximo es que, en el algoritmo de generación de farolillos, el sistema genera aproximadamente de manera uniforme alrededor del jugador en un arco de 360 grados. Según la teoría HRTF, el "ángulo mínimo discernible" es el más pequeño ángulo entre dos fuentes de sonido que el sistema auditivo puede distinguir. En un entorno RV, este ángulo puede ser más grande que en el mundo real. Tras varias pruebas, se determinó que un ángulo máximo de 36 grados entre diez objetos es un ángulo mínimo discernible adecuado. Bajo este campo en la figura 21, hay un toggle para Numero de objetos aleatorio, que, si se activa, permite al sistema generar un número aleatorio de farolillos entre el número establecido y 10.
- **Numero de rondas**: Define el número de rondas por nivel de dificultad. Cada acierto en localizar el farolillo correcto cuenta como una ronda completada. Tras finalizar todas las rondas, el juego avanza al siguiente nivel de dificultad (si lo hay). Este parámetro se ajusta mediante un método "*SetNumRound*" vinculado a la UI.
- **Distancia**: Este parámetro establece la distancia máxima vertical desde el eje del jugador hasta donde se pueden generar los objetos. El juego genera farolillos en una distancia aleatoria entre 1 metro y la distancia máxima desde el jugador. La presión sonora del farolillo disminuye con la distancia, por lo tanto, cuanto mayor sea este valor, mayor será la dificultad del entrenamiento. Este valor se ajusta junto con otros parámetros como altura, tiempo límite e intervalo de tiempo a través de un método "*SetValue*" vinculado a un slider en la UI.
- **Altura**: Define la altura máxima en la que se pueden generar los objetos. El juego genera farolillos entre la altura de la cabeza del jugador (cámara) y la altura máxima establecida. Establecer la altura mínima a la altura de la cabeza es para evitar que un

farolillo generado al azar aparezca directamente frente al jugador, lo que restaría significado al entrenamiento.

- **Tiempo límite para una ronda**: Limita el tiempo máximo para cada ronda. Si el jugador no localiza la fuente de sonido dentro de este tiempo, todos los farolillos generados se destruyen y se regeneran, haciendo que una reducción de este tiempo aumente significativamente la dificultad del juego.
- **Intervalo entre repeticiones de audio**: Decide la frecuencia con la que un farolillo sonoro reproduce su clip de audio, siendo el tiempo entre reproducciones consecutivas. Si se establece en 0 segundos, el audio se reproducirá en un bucle continuo. Aumentar este valor incrementa la dificultad de localizar la fuente de sonido.
- **Audios seleccionados para cada nivel**: El juego ofrece más de diez clips de audio cortos para localización, incluyendo voces de diferentes géneros y edades, sonidos de coches, entre otros. El jugador puede seleccionar uno o varios clips de audio, y en cada ronda, el objeto sonoro elegirá al azar uno de los audios seleccionados para reproducir. Estos sonidos de diferentes tonos ayudan a evaluar la capacidad auditiva del usuario en diferentes frecuencias.

Detalles de implementación de la selección de edición de audio: Los parámetros mencionados anteriormente y la interacción con la interfaz de usuario en su mayoría ya se han discutido en la introducción a la configuración de sonido. Solo el método para seleccionar audios utiliza un enfoque más complejo. Dado el gran número de audios y el deseo de integrar la capacidad de seleccionar múltiples audios, las opciones de usar múltiples toggles o un dropdown no eran adecuadas en la UI de Unity. Finalmente, se eligió el componente Scroll view, y se desarrollaron tres métodos interactivos para instanciar un componente Toggle para cada audio en la carpeta de recursos al iniciar el juego, y luego agregarlos al Scroll view.

El método PopulateToggles se encarga principalmente de cargar dinámicamente los clips de audio y crear los correspondientes interruptores de UI (Toggle). Estos interruptores permiten a los usuarios seleccionar los clips de audio específicos que desean usar en la terapia. Se utilizan los métodos "*Resources.LoadAll*" para cargar todos los recursos de audio del tipo AudioClip desde el directorio "Audio/Clips". Este método asegura que el juego pueda acceder a todos los archivos de audio predefinidos sin necesidad de codificarlos en el script. Para cada clip de audio cargado, se instanciará un interruptor (Toggle) predefinido y se agregará como hijo del scroll view. Cada interruptor lleva una etiqueta con el nombre del clip de audio correspondiente, lo que mejora la intuición y la interactividad de la interfaz de usuario. Al mismo tiempo, se añade un escuchador de eventos a cada interruptor, permitiendo que se realice una acción específica cuando el estado del interruptor cambia. Esto se logra mediante funciones anónimas, asegurando que cada acción del usuario desencadene la respuesta correcta.

Cuando los usuarios manipulan un interruptor de un clip de audio, **el método "*OnToggleValueChanged*"** se activa para manejar los cambios de estado del interruptor y actualizar las selecciones del usuario. Si el interruptor está activado (isOn), el método verifica si el clip de audio ya está incluido en la lista de selecciones del usuario. Si no está, se añade a

la lista; de lo contrario, si el interruptor está apagado, el clip de audio correspondiente se elimina de la lista. Al seleccionar un clip de audio, también se desencadena la reproducción de ese clip, proporcionando retroalimentación auditiva inmediata, útil para ajustes instantáneos en un entorno terapéutico.

El método "LoadSelectedToggles" se usa para cargar en la interfaz de usuario las selecciones de clips de audio previamente guardadas por el usuario. Este método recorre todos los interruptores y los compara con la lista de clips de audio seleccionados actualmente por el usuario. Si un clip de audio ha sido seleccionado por el usuario, el interruptor correspondiente se establecerá en estado activo; de lo contrario, se configurará como desactivado. Este proceso de sincronización asegura que la interfaz de usuario refleje la configuración actual del usuario, ya sea al iniciar sesión por primera vez o entre múltiples sesiones, manteniendo correctamente cargadas y mostradas las selecciones del usuario.

Implementación de diferentes configuraciones de dificultad: Otro diseño importante en esta fase permite a los usuarios establecer hasta cinco niveles de dificultad diferentes en la misma interfaz. Como se puede ver en la figura 22, en la esquina superior izquierda de la interfaz de usuario, se ha colocado un componente dropdown, que permite a los usuarios elegir entre Nivel1 y Nivel5 y en la figura de la derecha el juego se ha cambiado a la configuración del nivel 2. Al seleccionar cualquier nivel, el sistema llama al método "*LevelChanged*", que actualiza en la UI los ajustes del nivel de dificultad correspondiente almacenados en el archivo JSON del usuario.

Además, debajo del componente dropdown, hay cuatro toggles para los Niveles 2 a 4, que han agregado escuchadores para cambios de valor. Al cambiar el valor, se llama al método "*OnDifficultyToggleChanged*", permitiendo a los usuarios activar o desactivar ciertos niveles de dificultad. Esta funcionalidad soporta que los usuarios elijan uno o varios niveles de dificultad para jugar, siendo el Nivel1 activado por defecto. Después de completar las rondas del Nivel1, el sistema evaluará si hay otros niveles activos en el CurrentUser, y si los hay, cargará la configuración de ese nivel y continuará el juego.

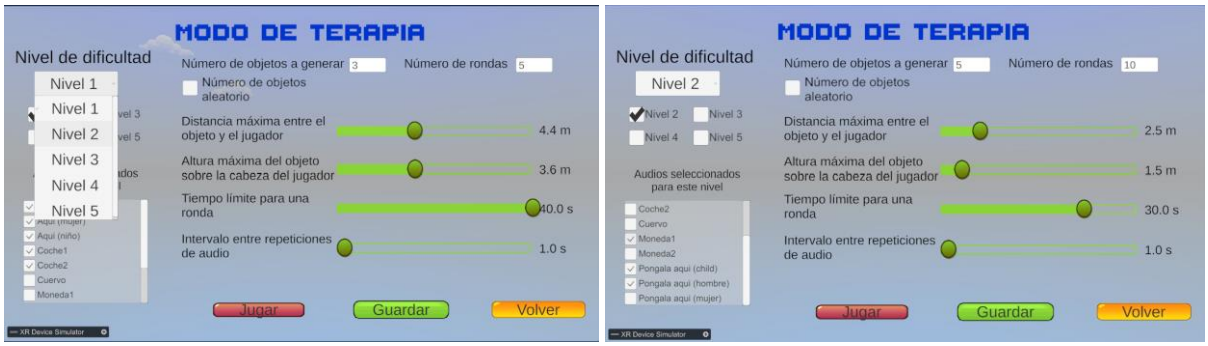


Figura 22: UI de la configuración del juego.

4.6 Lógica del juego

Introducción a la lógica principal de la mecánica del juego, enfocándose en la generación y gestión de objetos en posiciones aleatorias dentro del espacio del juego. Esta lógica emplea un diseño modular, utilizando principalmente tres scripts: "**ObjectGenerator.cs**", "**SkyLantern.cs**" y "**NonSkyLanternObject.cs**". La funcionalidad e interacción entre estos scripts se describe a continuación.

- **"ObjectGenerator.cs"**: Encargado de administrar todo lo relacionado con la generación de objetos (farolillos) y las mecánicas del juego. Este script calcula las posiciones de los objetos, genera los objetos, inicia corutinas de cuenta atrás con límite de tiempo, y maneja la lógica después de que los jugadores seleccionen el objeto de sonido correcto o incorrecto. Interactúa con los otros dos scripts para notificar a los objetos que reaccionen de manera correspondiente.
- **"SkyLantern.cs"**: Asignado a los farolillos voladores que emiten sonidos, este script gestiona su lógica. Estos farolillos son los principales objetivos que los jugadores necesitan identificar por la dirección del sonido en el juego. El script se encarga de la lógica de iluminación, ascenso y destrucción de los farolillos después de ser seleccionados.
- **"NonSkyLanternObject.cs"**: Aplicado a los farolillos que no emiten sonidos y sirven como elementos de distracción. Éste gestiona la lógica de destrucción de estos objetos.

4.6.1 "ObjectGenerator.cs"

Este componente se añade en la escena del juego a un objeto llamado GeneratorObj, un pilar de luz semitransparente (el área iluminada en la Figura 23), que tiene un colisionador en estado de "trigger". Cuando el jugador entra en el pilar de luz, éste desaparece y el juego comienza.

Figura 23: Zona de inicio del juego.

Inicialización y gestión de instancias: En el método "*Awake*" de "*ObjectGenerator.cs*", se inicializan los componentes necesarios, como el AudioSource que reproduce efectos de sonido y el componente Renderer del pilar de luz. También se obtiene el componente de movimiento del jugador para deshabilitar la movilidad del jugador una vez que el juego comienza. Este método asegura que los componentes esenciales estén listos antes de iniciar la lógica del juego. Además, verifica la existencia del objeto del jugador y sus componentes, emitiendo un mensaje de error si no se encuentran.

En el método Start, el sistema verifica la presencia del singleton "*SettingsManager*" en la escena. Si está presente, se obtiene el número de nivel de dificultad seleccionado por el jugador desde la instancia CurrentUser del "*SettingsManager*", y luego carga la configuración del juego para el Nivel1 del Generador de farolillos.

Generación de Farolillos: Esta es la parte central de la lógica del juego. A través del método "*GenerateObjects()*", el juego genera una cantidad determinada de objetos alrededor del jugador, uno de los cuales emitirá un sonido. El jugador necesita localizar este objeto basándose en la dirección del sonido. Aquí está la explicación detallada del método.

Activación del juego: Usando un método llamado "*OnTriggerEnter*" que toma un Collider como parámetro, el sistema determina si el jugador ha ingresado al área del juego (el pilar de luz azul). Una vez dentro, este método deshabilita el movimiento del jugador y llama a "*GenerateObjects*".

Determinación del número de generaciones: Después de que se llama a "*GenerateObjects*", el método primero verifica si se generan objetos en un número aleatorio. Si randomNumObj es verdadero, entonces usa "*Random.Range*" para seleccionar aleatoriamente la cantidad de objetos a generar dentro de un rango preestablecido (desde el número de generaciones configurado hasta 10). De lo contrario, usa el número preestablecido numObjects para la cantidad de generaciones.

Inicialización del arreglo de objetos generados: A continuación, el método inicializa un arreglo llamado generatedObjects para almacenar los objetos generados. Simultáneamente, inicializa la lista currentGeneratedObjects para gestionar las instancias de objetos actualmente generadas.

Determinación de la posición y rotación del jugador: Si el número de objetos generados es mayor que cero, el método obtiene la posición actual centerPosition y la orientación centerRotation de la cámara del jugador. Esta información se utilizará para calcular la posición y rotación específicas de los objetos generados.

Determinación del índice del farolillo con sonido: El método usa "*Random.Range*" para seleccionar aleatoriamente un índice dentro del rango del número de objetos generados, skyLanternIndex, el cual corresponderá al farolillo con sonido, mientras que los otros objetos generados serán elementos de distracción que no emiten sonido, aumentando así la aleatoriedad de la posición del objeto sonoro.

Configuración del rango de generación: Primero, el método define el rango específico de posiciones en las que se generará el farolillo alrededor del jugador, basándose en las distancias mínimas y máximas minDistance y maxDistance, así como las alturas mínimas y máximas minHeight y maxHeight, mencionadas en la sección de configuración del mecanismo.

Prevención de la superposición de posiciones: Aunque la configuración del rango de generación asegura la aleatoriedad de las posiciones de los objetos, también introduce el problema de que los objetos generados pueden superponerse en posición, ya sea una superposición completa o una proximidad muy cercana, lo que puede impedir que los pacientes con DPAC e incluso las personas con sistemas auditivos normales distingan correctamente las fuentes de sonido. Para resolver este problema, se usa un "*HashSet<float>"* para crear un conjunto usedAngles, que es un conjunto que garantiza la unicidad de sus elementos, por lo tanto, es muy adecuado para almacenar y verificar los ángulos ya utilizados. El método específico es definir primero una diferencia de ángulo mínimo minAngle, que determina el intervalo angular mínimo entre un objeto generado nuevo y los objetos ya existentes. En cada generación de un objeto, se genera un ángulo aleatorio entre 0 y 360 grados, utilizando el método "*Mathf.DeltaAngle*" para calcular en un bucle el valor absoluto de la diferencia entre el nuevo ángulo y todos los ángulos ya almacenados en el HashSet, asegurando que esta diferencia sea mayor que el minAngle establecido. Si la diferencia de ángulo es menor que minAngle, se regenera un ángulo aleatorio hasta encontrar uno que cumpla con la condición. Una vez encontrado un ángulo adecuado, se añade al conjunto usedAngles, y luego, basándose en este ángulo, se calcula la posición y rotación del objeto. La rotación se convierte en ángulo, y la posición del objeto se calcula basándose en la ubicación del jugador.

Generación cíclica de objetos: Con un método ya establecido para calcular posiciones aleatorias que no se superponen, el método procede a un bucle for que se repite tantas veces como objetos se deban generar. En cada iteración del bucle:

1. Dentro del rango de generación mencionado anteriormente, se generan una distancia aleatoria currentDistance y una altura currentHeight.
2. Se genera un ángulo angle de manera aleatoria y se asegura que este ángulo tenga una diferencia suficiente con todos los ángulos ya utilizados.
3. El nuevo ángulo se añade al conjunto usedAngles.
4. Basándose en el ángulo, se calcula la rotación del objeto rotation, y a partir de rotation y currentDistance se determinan las coordenadas en el eje x y z de la posición position, mientras que la coordenada en el eje y es igual a currentHeight. De esta manera, en cada ciclo se determina la posición de cada farolillo generado respecto al jugador.
5. Según el índice de farolillo con sonido generado al azar antes del ciclo, se instancia el objeto correspondiente. Si el índice corresponde a un farolillo con sonido, entonces se instancia skyLanternPrefab (un prefabricado de farolillo con un componente de audio source) y se asocia su componente SkyLantern con ObjectGenerator; de lo contrario, se instancia NonSkyLanternPrefab (un prefabricado de farolillo sin componente de audio source). La Figura 24 muestra un ejemplo de farolillos generados en una ronda

del juego, donde se puede observar que todos los farolillos tienen una apariencia idéntica.

Figura 24: Un juego en curso con tres farolillos voladores generados.

Restricción del tiempo de selección mediante "Coroutines": Una vez finalizado el bucle y generada la cantidad establecida de farolillos, el juego pasa a la fase en la que los jugadores deben localizar la fuente de sonido. Esta fase impone un límite de tiempo para la localización del jugador utilizando una técnica especial en Unity llamada "*Coroutines*", que permite pausar y reanudar la ejecución del hilo a lo largo de varios frames, facilitando operaciones de retraso no bloqueantes. En el desarrollo de juegos, las "*Coroutines*" se usan frecuentemente para implementar temporizadores, animaciones y operaciones asíncronas. En este proyecto, se utilizan extensivamente para controlar la reproducción de audio y los retrasos en la generación de objetos.

El tipo de retorno de un método "*Coroutine*" debe ser IEnumerator, por lo que al definir el método "*Coroutine*" se escribe "*private IEnumerator SelectionTimeLimit()*". Dentro de este método, se define un temporizador que incrementa cada frame y sigue contando mientras sea menor que "*selectionTimeLimit*". Si el jugador localiza la fuente de sonido dentro del tiempo limitado, el temporizador se detiene. Si el jugador no elige correctamente el farolillo antes de que el tiempo se agote, se destruyen los objetos y se reproduce un sonido indicativo de que el tiempo ha expirado. A continuación, se inicia otra "*Coroutine*", "*RegenerateObjectsWithDelay*", que está diseñada para regenerar objetos después de un tiempo de retraso específico, permitiendo un breve descanso entre rondas. Esta "*Coroutine*" espera un tiempo determinado antes de volver a llamar al método "*GenerateObjects*" para reiniciar la ronda.

Acciones tras seleccionar un farolillo: Cuando se selecciona un farolillo, se llama a uno de los métodos, "*OnSkyLanternSelected*" si la elección es correcta o "*OnNonSkyLanternObjectSelected*" si es incorrecta. El primero, al elegir correctamente, reproduce un sonido de reconocimiento, actualiza el contador de rondas jugadas y verifica si se debe avanzar al siguiente nivel de dificultad. Además, destruye todos los objetos no seleccionados y prepara la generación de nuevos objetos para la siguiente ronda, siempre que no se haya alcanzado el límite de rondas. El segundo, en caso de una elección incorrecta, reproduce un sonido de error y actualiza el contador de elecciones incorrectas.

Detención del juego: El juego de localización de farolillos seguirá repitiéndose hasta completar todas las rondas establecidas para el Nivel1. Luego, el sistema leerá la configuración del siguiente nivel de dificultad establecido por el usuario desde el objeto CurrentUser (si existe) y comenzará un nuevo juego con esa configuración de dificultad, hasta completar todas las rondas de todos los niveles de dificultad. El Generador dejará de generar objetos una vez completadas todas las rondas, y el sistema preguntará al jugador si desea ver los resultados de la sesión de juego.

4.6.2 "SkyLantern.cs" y "NonSkyLantern.cs"

Ya se ha explicado la implementación de la mecánica principal del juego. En este capítulo, se explicará cómo "*ObjectGenerator.cs*" interactúa con los farolillos con y sin sonido, así como algunos aspectos del diseño de estos dos tipos de farolillos.

Carga de Configuración:

Dado que "*SkyLantern.cs*" se añade a los farolillos que emiten sonido, también necesita cargar el audio que se reproducirá en el componente audio source. A través del método "*LoadLanternSettings*", cada farolillo con sonido carga un audio aleatorio de la configuración del usuario cuando se genera.

Interacción del Usuario:

Ambos tipos de farolillos tienen el componente XRSimpleInteractable en sus prefabricados. Este componente, proporcionado por Unity, permite que los objetos sean interactuables directamente por los usuarios con dispositivos XR, soportando operaciones básicas como tocar y agarrar. En este proyecto, este componente es esencial para que los jugadores puedan seleccionar los objetos que emiten sonido. Al añadir diferentes listeners al componente XRSimpleInteractable en los métodos "*start()*" de los scripts "*SkyLantern.cs*" y "*NonSkyLantern.cs*", se permite que los usuarios interactúen con los farolillos de distintas maneras, incluyendo:

- **Listener del evento activated**: Se activa el método "*OnActive*" cuando el usuario selecciona un farolillo. Este método maneja la lógica de selección del farolillo. Para el farolillo con sonido, se activa su componente de luz, emitiendo luz, se detiene la reproducción de audio y el farolillo comienza a ascender de manera uniforme hasta que se destruye a cierta altura. Finalmente, se llama al método "*OnSkyLanternSelected*"

del objectGenerator, notificando al generador que el jugador ha seleccionado el objeto que emite sonido, permitiendo al ObjectGenerator realizar las operaciones subsiguientes. La Figura X muestra un farolillo ascendiendo. En el caso de los farolillos sin sonido, tras ser activados, se autodestruyen y notifican al ObjectGenerator. Este diseño asegura una respuesta y retroalimentación inmediatas a la acción del usuario, mejorando la interactividad y la experiencia del usuario en el juego.

- **Listeners de los eventos "hoverEntered" y "hoverExited"**: Además, ambos scripts añaden estos listeners para manejar la interacción cuando el rayo del usuario apunta o se aleja de un farolillo. Cuando el rayo del usuario apunta a un farolillo, el sistema llama al método "*OnHoverEnter*", ajustando la transparencia del farolillo a opaco para indicar la selección actual del usuario. Por el contrario, cuando el rayo se aleja del farolillo, el sistema llama al método "*OnHoverExited*", restaurando la transparencia a su estado semitransparente. En la Figura 25, se puede observar en la figura de la izquierda que cuando el rayo apunta a uno de los farolillos, la transparencia de ese farolillo es claramente diferente de los demás. Estas retroalimentaciones visuales ayudan al usuario a interactuar de manera más intuitiva, mejorando la jugabilidad y la inmersión en el juego.

Figura 25: Farolillo cambia de semitransparente a opaco al seleccionarlo (izquierda). Tras seleccionar el farolillo sonoro, se enciende y se eleva al cielo (derecha).

Por último, para visualizar claramente el funcionamiento de la mecánica del juego, el diagrama de flujos de la mecánica del juego se muestra en la Figura 26.

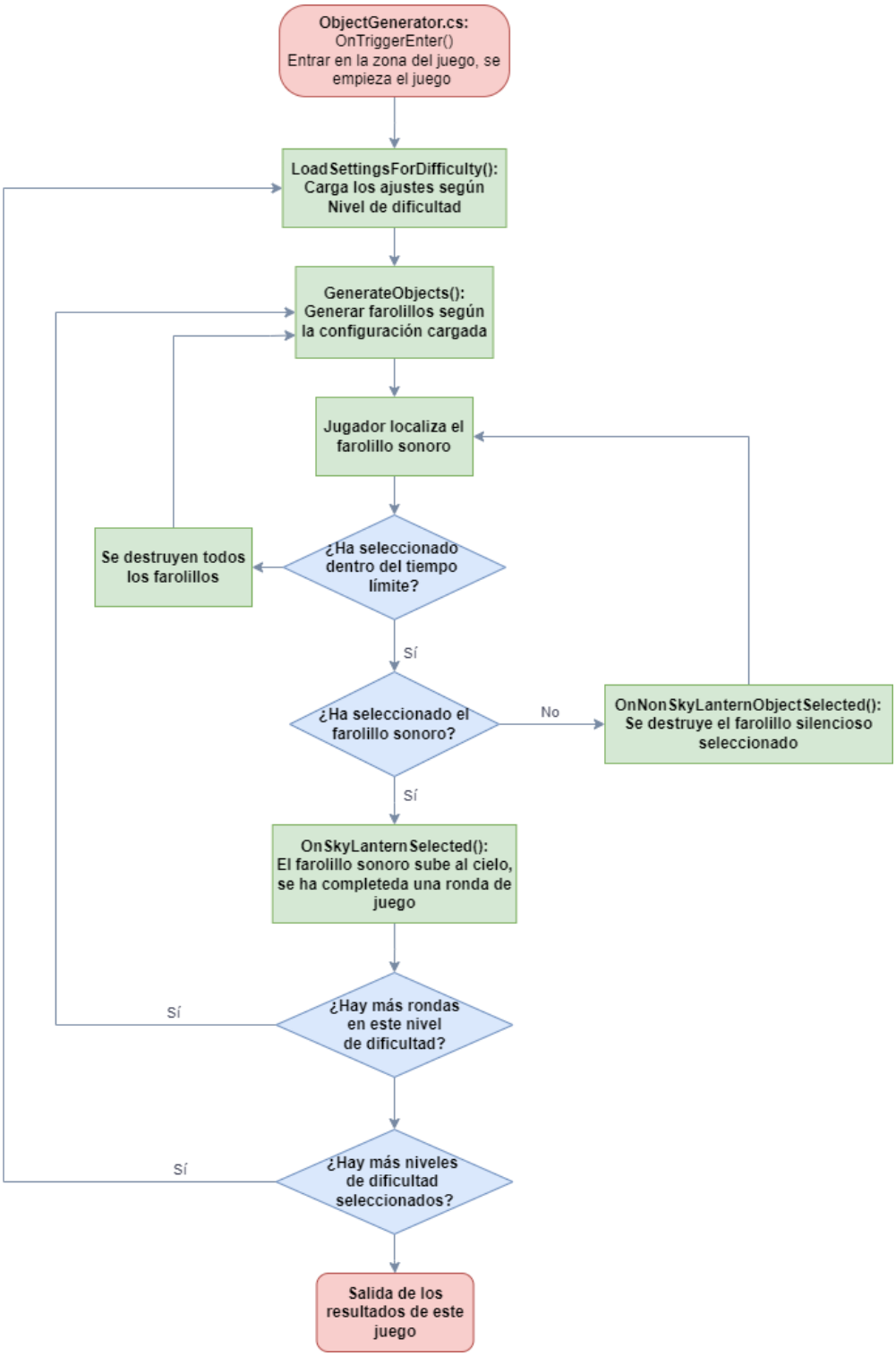


Figura 26: Diagrama de flujo de la mecánica del juego.

4.7 Salida de resultados del juego

Registro y Exportación de Resultados del Juego

Como un videojuego terapéutico, además de las mecánicas de entrenamiento del juego, la función de salida de resultados es indispensable. En esta sección, se detallará cómo registrar y exportar los resultados del juego del jugador. Utilizando el script "*OutputManager*", el juego puede guardar la configuración del usuario y los resultados del juego en un archivo CSV. Este método se ejecuta automáticamente una vez que el jugador completa todas las rondas de todos los niveles de dificultad.

Inicialización de la Salida:

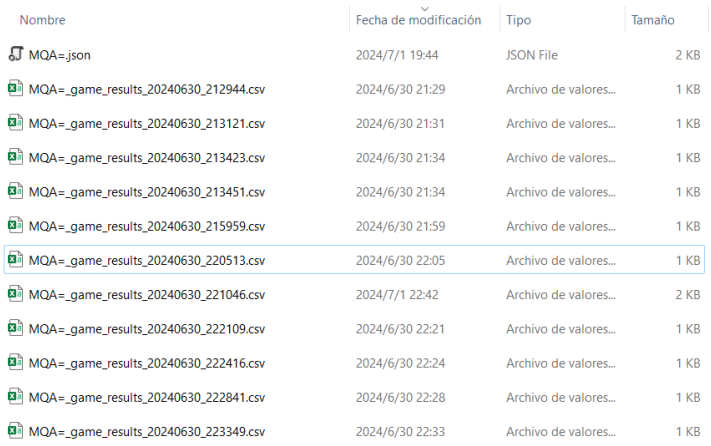
Una de las funciones clave del script "*OutputManager*" es inicializar el proceso de salida y asegurar que el nombre del archivo generado sea único cada vez. Al agregar una marca de tiempo al nombre del archivo, se garantiza la unicidad del mismo, evitando que se sobrescriban archivos. El método "*InitializeOutput(User currentUser)*" se llama al inicio del juego, generando un nombre de archivo único y llamando al método "*SaveUserSettingsToCsv*" para guardar primero todas las configuraciones del usuario para ese juego.

Creación del Archivo y Escritura de Información Básica:

El método "*SaveUserSettingsToCsv*" se encarga de escribir todas las configuraciones actuales del usuario en un archivo CSV. Estas configuraciones incluyen el nombre de usuario, varios ajustes de volumen y configuraciones de juego para diferentes niveles de dificultad. Utiliza "*StreamWriter*" para crear y escribir las configuraciones básicas del usuario, incluyendo el nombre de usuario y los ajustes de volumen.

Guardar Resultados del Juego:

El método "*SaveResultsToCsv*" se encarga de añadir los resultados específicos del juego al archivo CSV. El método recibe una lista de resultados del juego y el nivel de dificultad seleccionado, y utiliza "StreamWriter" en modo de adición para escribir estos resultados en el archivo. Los resultados para cada nivel de dificultad incluyen el número de la ronda, el nombre del fragmento de audio, el número de selecciones incorrectas y el tiempo total. De esta manera, se guarda un registro completo del juego, facilitando el análisis posterior por parte del jugador y los investigadores. La Figura 27 muestra una serie de archivos de resultados con marcas de tiempo generados por un usuario en su sistema local. Las Tablas 3 y 4 presentan los resultados de una sesión de juego, la primera mostrando las configuraciones del juego para dos niveles de dificultad y la segunda registrando los resultados de cada ronda para esos niveles.



Nombre	Fecha de modificación	Tipo	Tamaño
MQA=.json	2024/7/1 19:44	JSON File	2 KB
MQA=_game_results_20240630_212944.csv	2024/6/30 21:29	Archivo de valores...	1 KB
MQA=_game_results_20240630_213121.csv	2024/6/30 21:31	Archivo de valores...	1 KB
MQA=_game_results_20240630_213423.csv	2024/6/30 21:34	Archivo de valores...	1 KB
MQA=_game_results_20240630_213451.csv	2024/6/30 21:34	Archivo de valores...	1 KB
MQA=_game_results_20240630_215959.csv	2024/6/30 21:59	Archivo de valores...	1 KB
MQA=_game_results_20240630_220513.csv	2024/6/30 22:05	Archivo de valores...	1 KB
MQA=_game_results_20240630_221046.csv	2024/7/1 22:42	Archivo de valores...	2 KB
MQA=_game_results_20240630_222109.csv	2024/6/30 22:21	Archivo de valores...	1 KB
MQA=_game_results_20240630_222416.csv	2024/6/30 22:24	Archivo de valores...	1 KB
MQA=_game_results_20240630_222841.csv	2024/6/30 22:28	Archivo de valores...	1 KB
MQA=_game_results_20240630_223349.csv	2024/6/30 22:33	Archivo de valores...	1 KB

Figura 27: Los archivos de resultados del juego.

Tabla 3: Las configuraciones guardadas en el archivo de resultado para un juego de Prueba.

User Settings	
Username	MQA=
Master Volume	0.8
BGM Volume	1
Feedback Volume	1
Objects Volume	1
Selected BGM Index	3
Difficulty 1 Settings	
Number of Objects	3
Distance	4.35046
Height	3.619262
Time Limit	40
Rounds to Play	4
Audio Interval	1
Selected Audio Clips	Aqui (niño);Coche1;Coche2;Aqui (mujer)
Difficulty 2 Settings	
Number of Objects	4
Distance	2.5
Height	1.5
Time Limit	30
Rounds to Play	5
Audio Interval	1
Selected Audio Clips	Aqui (niño);Aqui (mujer)

Tabla 4: Los resultados de un juego de Prueba como ejemplo.

Difficulty 1 Results			
Round	AudioClip	IncorrectSelections	TotalTime
Round1	Aqui (mujer)	1	9.040179
Round2	Coche2	2	6.793261
Round3	Coche2	2	9.040179
Round4	Coche1	1	6.793261

Difficulty 2 Results			
Round	AudioClip	IncorrectSelections	TotalTime
Round1	Aqui (mujer)	1	9.040179
Round2	Aqui (niño)	0	6.793261
Round3	Aqui (niño)	2	9.040179
Round4	Aqui (mujer)	3	6.793261
Round5	Aqui (niño)	0	6.793261

A través del script "*OutputManager*", el juego puede registrar detalladamente las configuraciones y los resultados del juego del jugador. Esto no solo ayuda a los jugadores a comprender su rendimiento en el juego, sino que también proporciona valiosos recursos de datos para los investigadores, permitiéndoles analizar el efecto del juego en la rehabilitación de niños con DPAC. Además, el formato CSV generado es universal y fácil de integrar con otras herramientas de análisis de datos, lo que aumenta aún más el valor de los datos recopilados.

5 Resultados

Este capítulo aborda las pruebas realizadas durante el desarrollo del proyecto y después de su finalización para verificar la funcionalidad y efectividad del proyecto.

Primero, durante el desarrollo del proyecto, después de implementar el mecanismo de generación de objetos sonoros y no sonoros, se realizaron pruebas con otros desarrolladores de "El planeta Sonoa" (estudiantes responsables de otros módulos y la tutora) y varios terapeutas auditivos de Aurea TAV. Estas pruebas preliminares verificaron la precisión del audio espacial. A través de múltiples pruebas de localización de sonidos en el juego por individuos con sistemas auditivos normales, los evaluadores lograron localizar la fuente sonora con una precisión superior al 90%. En esta etapa de pruebas, la evaluación subjetiva profesional de los terapeutas de DPAC fue positiva respecto a la funcionalidad del proyecto.

Tras completar el proyecto, se diseñaron cinco niveles de dificultad progresivos para validar la flexibilidad y adaptabilidad del videojuego terapéutico para pacientes en diferentes condiciones. Se realizaron pruebas con un grupo pequeño de adolescentes con audición normal que no estaban familiarizados con el proyecto. Los resultados mostraron que el juego tiene una efectividad notable en la ajustabilidad de la dificultad.

Además, se consideró diseñar experimentos para validar el impacto de audios con diferentes tonos y diferentes ruidos de fondo en los resultados del entrenamiento, pero debido a limitaciones de tiempo, no se llevaron a cabo estas pruebas.

En resumen, las pruebas anteriores demostraron un éxito inicial del proyecto en alcanzar su objetivo principal de desarrollar un videojuego terapéutico RV centrado en la localización de sonidos en un espacio 3D. Sin embargo, debido a la falta de pruebas con grupos de control, no se pudo verificar la efectividad del proyecto en la rehabilitación de pacientes con DPAC y en personas con audición normal.

6 Presupuesto

En esta sección se resumen los costes implicados en la realización de este proyecto, ya que el proyecto está diseñado para el desarrollo de un videojuego de RV, un casco de RV es una necesidad, el equipo de RV utilizado en este proyecto fue alquilado por el equipo GAMMA pero debe ser considerado a su coste original, el ordenador necesario para el desarrollo del videojuego tiene que ser capaz de cumplir con los requisitos mínimos de hardware del equipo de RV utilizado, de lo contrario surgirán problemas durante el proceso de desarrollo. Es necesario que el ordenador cumpla al menos los requisitos mínimos de hardware del equipo de VR utilizado, de lo contrario surgirán problemas durante el proceso de desarrollo, y se incurrirá en costes por el software, los modelos, el audio y otros periféricos informáticos como el ratón. Por último, también hay que calcular los costes de mano de obra, como el salario del desarrollador. A continuación, se describe detalladamente cada partida de costes:

- Ordenador portátil: HP OMEN 16-C0042NS con procesador de AMD Ryzen 7 5800H, memoria RAM de 16GB, almacenamiento de 512GB SSD y un GPU de modelo RTX 3050 Ti/16.1" (949 €).
- Ratón de Logitech G203 (26,8 €).
- Auriculares: Apple EarPods (19 €).
- Gafas de realidad virtual con controladores de movimiento: Lenovo Explorer [40](399 €).
- Softwares utilizados: Unity Engine 2021.3.18.f1, Blender, Audacity, Visual Studio (Gratis).
- Recursos utilizados: 1. Audios clips descargados en la página web de FreeSound [48]. 2. SkyBox, UI Sprite, Modelos decorativos descargados en Unity Asset Store (Gratis).
- En cuanto a los costes laborales, según los resultados de una búsqueda, el salario medio mensual de un desarrollador de videojuegos junior en España es de 1.333€ [49], Al tratarse de un proyecto con unos tres meses (480 horas) de trabajo, el coste de la mano de obra se calculará multiplicando el salario medio por tres meses, lo que equivale a 3999 €.

Junto con los costes anteriores, se ha elaborado una tabla de costes del proyecto：

Tabla 5: Presupuesto estimado del proyecto.

Concepto	Unidades	Coste unitario	Coste Total
Material			
Ordenador portátil	1	949,00 €	949,00 €
Ratón	1	26,80 €	26,80 €
Gafas de RV	1	399,00 €	399,00 €
Auriculares	1	19,00 €	19,00 €
Total parcial			1393,80 €
Softwares y recursos			
Unity	1	0,00 €	0,00 €
Blender	1	0,00 €	0,00 €
Visual Studio	1	0,00 €	0,00 €
Todos recursos	varios	0,00 €	0,00 €
Total parcial			0,00 €
Personal			
Desarrollador de videojuegos junior	1	1333,00 €/mes	3999,00 €
Total			5392,80 €

Según las estadísticas, el coste total del proyecto se estima en unos 5.392,8 euros, y los principales costes proceden de los salarios de los desarrolladores del juego y de los equipos de hardware, como ordenadores y gafas RV, pero los costes personales representan el 74% del coste total, y los costes de material pueden reducirse comprando equipos de segunda mano, etc., lo que basta para demostrar que los costes necesarios para realizar este proyecto son mínimos, y que, al tratarse de un proyecto de videojuegos terapéuticos de utilidad pública, el coste podría considerarse razonable.

7 Impacto del proyecto

7.1 Impacto ambiental

Al desarrollar un videojuego de terapia de RV, el proyecto contribuye indirectamente al medio ambiente al reducir el uso de otros grandes equipos utilizados en los métodos terapéuticos tradicionales. Aunque la adquisición de equipos de RV puede resultar inicialmente perjudicial para el medio ambiente en comparación con los métodos tradicionales de entrenamiento de las habilidades de diagnóstico y localización de sonidos (que requieren un mínimo de cuatro equipos, como ordenadores o altavoces colocados alrededor del paciente), el uso de sonido 3D suficientemente realista utilizado en los equipos de RV simplifica el uso de los equipos (Coincide con el ODS 12: Producción y Consumo Responsables) [50].

7.2 Impacto social

El impacto social principal del proyecto se manifiesta en los siguientes aspectos:

- **Mejora de la calidad de vida de los pacientes**: A través de un juego divertido e interactivo, este proyecto ayuda a aumentar la inmersión y la participación de los niños en el entrenamiento auditivo, mejorando así su capacidad de procesamiento auditivo. Esto mejorará significativamente su calidad de vida, especialmente en sus estudios del futuro (ODS 3: Salud y Bienestar).

- **Apoyo a padres y terapeutas**: El proyecto proporciona a los terapeutas una nueva y efectiva herramienta para ayudar en su trabajo, y ofrece a los padres un método para realizar terapias divertidas en casa, así se alivia la carga de ellos.

- **Educación**: La promoción de este tipo de juegos serios puede aumentar la conciencia pública sobre el DPAC, fomentando la comprensión y el apoyo social hacia este grupo (ODS 4: Educación de calidad y ODS 10: Reducción de las desigualdades) [50].

7.3 Impacto académico

Las contribuciones académicas del proyecto incluyen (ODS 9: Innovación) [50]:

- **Integración multidisciplinaria**: El proyecto combina diseño de videojuegos, ingeniería de sonido, tecnología de RV y terapias, mostrando un gran potencial de la investigación y aplicación interdisciplinaria.

- **Recursos educativos innovadores**: El proyecto ofrece a las instituciones educativas un caso de estudio vívido, demostrando cómo aplicar tecnologías avanzadas en el campo médico y de rehabilitación, ayudando a inspirar el pensamiento innovador y las habilidades prácticas de los estudiantes.

- **Expansión del proyecto**: Este proyecto puede ampliarse para desarrollar más proyectos relacionados con el audio espacial y la localización de sonidos.

8 Conclusiones

En este capítulo se revisará todo el proceso de desarrollo del proyecto y los resultados obtenidos desde la perspectiva de la formulación de los principales objetivos del proyecto, y se resumirán las dificultades y revelaciones del proceso de desarrollo.

8.1 Revisión del proyecto

El objetivo principal de este proyecto es desarrollar un videojuego terapéutico de realidad virtual (RV) que ayude a los niños con DPAC a ejercitar su capacidad de localizar sonidos en un espacio 3D, como complemento del proyecto "El planeta Sonoa". Al diseñar la mecánica del juego, se consideró que, además de cumplir con la tarea de entrenamiento, el juego debía ser lo suficientemente atractivo para mantener la atención de los niños y mitigar el aburrimiento y la frustración asociados con la naturaleza repetitiva de la rehabilitación. El mayor desafío inicial fue equilibrar la utilidad y la diversión del juego.

Después de varias sesiones de lluvia de ideas y discusiones en equipo, se decidió combinar la localización de sonidos en un espacio 3D con una tradición del Año Nuevo chino, lanzar farolillos voladores, una idea que surgió al reflexionar sobre la falta de elementos culturales en "El planeta Sonoa". Este proyecto ya incluye diversas características geográficas y culturales, ofreciendo una experiencia más fresca y envolvente para los usuarios, y la adición de un módulo cultural asiático enriquecería aún más el proyecto. Así, la mecánica del juego se estableció: los jugadores, en un pueblo de arquitectura china donde viven pandas, ayudan a los pandas a identificar el "verdadero" farolillo que emite sonido entre varios "falsos" que no lo emiten.

Una vez diseñada la mecánica del juego, el siguiente paso fue considerar cómo diseñar diferentes parámetros ajustables para aumentar la adaptabilidad del juego, permitiendo una mayor flexibilidad para pacientes en diversas condiciones. Tras múltiples reuniones y pruebas con la tutora Martina Eckert, el desarrollador de servidores web Miguel Jiménez Arévalo y los estudiantes responsables de otros módulos, Pablo Lucas Olivares, Bogdán Morar y Mario Ortega García, se decidió controlar la dificultad del entrenamiento mediante la modificación de parámetros como el número de farolillos generados, el rango de generación y los audios de localización.

Luego, dado que este proyecto no se integró en "El planeta Sonoa", se añadieron nuevos módulos para la gestión de usuarios y la salida de resultados, funciones igualmente importantes en un videojuego terapéutico, proporcionando las condiciones básicas para la configuración personalizada y la evaluación de la efectividad del tratamiento.

Finalmente, se dedicó tiempo al diseño estético de la UI y los escenarios del juego, que, aunque no son la parte central de la solución, contribuyen significativamente a la inmersión y al disfrute del juego, elementos esenciales en un videojuego terapéutico completo.

8.2 Dificultades y logros del proceso de desarrollo

Primero, en la fase inicial del desarrollo, se encontró un problema de incompatibilidad entre el dispositivo de RV utilizado y varias computadoras. Resolver este problema requirió mucho tiempo, incluyendo la verificación de la compatibilidad del sistema operativo y la integridad de los complementos del dispositivo. Finalmente, se determinó que la causa era que el hardware de la computadora, especialmente la GPU, no cumplía con los requisitos mínimos para utilizar las gafas de RV Lenovo Explorer. Este proceso destacó la importancia de la preparación en el desarrollo de proyectos, enfatizando la necesidad de verificar la disponibilidad de todos los equipos necesarios y de realizar pruebas de control de variables para identificar problemas.

En cuanto a la conceptualización de la solución del proyecto, determinar la dirección del proyecto fue un proceso complejo. En la fase inicial, se consideró otra idea para la mecánica del juego que también incorporaba la cultura del Año Nuevo chino. Según una antigua leyenda, los chinos lanzaban petardos en el Año Nuevo para ahuyentar a una criatura llamada Nian. En este diseño inicial, los jugadores debían localizar a Nian escuchando sus sonidos y luego lanzar petardos para ahuyentarlo. Sin embargo, esta idea fue descartada porque el contenido de violencia y monstruos no era adecuado para niños pequeños. Esto subraya la importancia de considerar cuidadosamente todos los aspectos al proponer soluciones, incluyendo la aceptabilidad para el grupo objetivo.

Además, las discusiones periódicas con otros desarrolladores y las pruebas mutuas fueron cruciales para mantener el proyecto en la dirección correcta. Trabajar solo puede llevar a una mentalidad rígida, donde es difícil identificar defectos de diseño y problemas diversos. Las diferentes perspectivas y enfoques de cada individuo son inmensamente beneficiosos para perfeccionar el diseño del proyecto.

Durante el desarrollo del proyecto, se encontraron desafíos técnicos específicos. Por ejemplo, en la implementación del mecanismo de generación de objetos, se probaron varios métodos. Con un método completamente aleatorio, los objetos a veces se generaban muy cerca o incluso se superponían, lo que dificultaba la identificación de la fuente de sonido por parte del jugador. Por otro lado, un método regulado hacía que el juego perdiera desafío, ya que los objetos se generaban de manera uniforme alrededor del jugador. Finalmente, se optó por el uso de "*HashSet*", un método semi-aleatorio (determinando un ángulo mínimo entre los objetos, mientras que otros parámetros se generan aleatoriamente). Aunque este método equilibra el desafío, ocasionalmente genera objetos muy cercanos en algunas rondas.

Finalmente, en cuanto a la efectividad del proyecto para el entrenamiento de rehabilitación, faltaron validaciones rigurosas. Solo se verificó la precisión de la localización del sonido en el espacio 3D y la diversión del juego con evaluadores de audición normal, sin realizar pruebas con un grupo de control de pacientes con DPAC.

9 Trabajos futuros

A la luz de las conclusiones de la sección anterior, en este capítulo se proyectan las futuras posibilidades de expansión y mejora del proyecto:

1. **Diseño Estético del Juego**: Debido a las limitaciones de tiempo, se utilizaron métodos básicos y muchos recursos gratuitos de la tienda de modelos para el diseño de escenarios y personajes. Para aumentar la inmersión del juego en el futuro, se podría optimizar esta parte diseñando más modelos que se ajusten al contexto cultural y añadiendo animaciones a los modelos de pandas.
2. **Métodos de Gestión de Usuarios**: Actualmente, se utiliza un método muy básico de encriptación de nombre de usuario y contraseña, que solo impide que el nombre de usuario sea detectado en el directorio del sistema. Sin embargo, si el nombre de usuario o el directorio del sistema donde se almacenan los archivos JSON de los usuarios se expone, se puede descifrar fácilmente usando métodos de encriptación y desencriptación disponibles en Internet o en cualquier lenguaje de programación. En el futuro, sería ideal reemplazar este método por uno que ofrezca una verdadera función de encriptación.
3. **Implementación de la Mecánica del Juego**: Actualmente se utiliza un método de generación de objetos con un alto grado de aleatoriedad. En el futuro, se podría integrar otro método más regulado como un parámetro adicional de dificultad del juego, permitiendo a los jugadores personalizar su experiencia y aumentando la flexibilidad del juego.
4. **Modo de Historia**: Durante el desarrollo del proyecto, se consideró la posibilidad de desarrollar un modo de juego adicional, el modo historia. En este modo, los jugadores seguirían una ruta establecida en el juego, completando gradualmente diferentes tareas de localización de sonido mientras interactúan con los pandas. Este modo no se implementó debido a la limitación de tiempo, pero podría incluirse en futuras expansiones.
5. **Función para añadir y eliminar audio**: Para el entrenamiento de posicionamiento de sonido, es muy significativo intentar utilizar diferentes frecuencias, diferentes longitudes y diferentes tonos de audio para el entrenamiento, lo que puede entrenar la capacidad del jugador para localizar el sonido en diferentes escenarios. En el futuro, se puede integrar una función en el proyecto para añadir o eliminar audio opcional en el juego modificando el audio en el directorio local, de modo que el terapeuta y el usuario puedan ampliar el contenido del entrenamiento de acuerdo con su propia situación.
6. **Función de Salida de Resultados del Juego**: Actualmente, además de la configuración del juego, solo se pueden registrar el número de errores por ronda, el audio reproducido en ese momento y el tiempo empleado en encontrar la fuente sonora correcta. Si se añaden más resultados evaluables, se podría ayudar más a los terapeutas en la evaluación de los pacientes. Por ejemplo, se podría diseñar un método para registrar la posición de los objetos correctos e incorrectos en relación con el

jugador, permitiendo identificar en qué direcciones el jugador tiene más dificultades para localizar sonidos.

7. **Eficacia del Entrenamiento**: Es una lástima que no se haya diseñado ni realizado un experimento riguroso para evaluar la eficacia del entrenamiento del juego. En el futuro, si se tiene la oportunidad de colaborar con instituciones relevantes, se podrían establecer experimentos controlados estrictos, dividiendo a un número suficiente de pacientes infantiles con DPAC en dos grupos. Uno de los grupos usaría este proyecto para realizar entrenamientos de rehabilitación durante un tiempo prolongado, mientras que el otro grupo no usaría el juego o utilizaría otros juegos para entrenarse. Luego, se podrían comparar las habilidades de localización de sonido entre los niños entrenados y los no entrenados, verificando así la efectividad del proyecto.

10 Referencias

[1] CITSEM UPM, «Grupo de Aplicaciones Multimedia y Acústica (GAMMA)». Accedido: 25 de junio de 2024. [En línea]. Disponible en: https://www.citsem.upm.es/es/quienes-somos/grupos-de-investigacion/grupo-de-aplicaciones-multimedia-y-acustica-gamma

[2] G. D.; H. Chermak James W. III; Musiek Frank E., «Differential Diagnosis and Management of Central Auditory Processing Disorder and Attention Deficit Hyperactivity Disorder», *J Am Acad Audiol*, vol. 10, n.º 06, pp. 289-303, 1999, doi: 10.1055/s-0042-1748501.

[3] P. Dawes, D. V. M. Bishop, T. Sirimanna, y D.-E. Bamiou, «Profile and aetiology of children diagnosed with auditory processing disorder (APD)», *Int J Pediatr Otorhinolaryngol*, vol. 72, n.º 4, pp. 483-489, 2008, doi: https://doi.org/10.1016/j.ijporl.2007.12.007.

[4] O. Cañete S, «Desorden del procesamiento auditivo central (DPAC)», *Revista de otorrinolaringología y cirugía de cabeza y cuello*, vol. 66, pp. 263-273, 2006, Accedido: 23 de junio de 2024. [En línea]. Disponible en: http://dx.doi.org/10.4067/S0718-48162006000300014.

[5] S. Hind, «Survey of care pathway for auditory processing disorder», *Audiol Med*, vol. 4, n.º 1, pp. 12-24, 2006.

[6] M. P. Masquelier, «Management of auditory processing disorders.», *Acta Otorhinolaryngol Belg*, vol. 57, n.º 4, pp. 301-310, 2003.

[7] J. Weihing, G. Chermak, y F. Musiek, «Auditory Training for Central Auditory Processing Disorder», *Semin Hear*, vol. 36, pp. 199-215, oct. 2015, doi: 10.1055/s-0035-1564458.

[8] I. Granic, A. Lobel, y R. C. M. E. Engels, «The benefits of playing video games.», *American psychologist*, vol. 69, n.º 1, p. 66, 2014, doi: https://doi.org/10.1037/a0034857.

[9] European Games Developer Federation (EGDF), Interactive Software Federation of Europe (ISFE), y Video Games Europe (VGE), «2022 All About Video Games – European Key Facts», 2022.

[10] J. Piaget, *Play, Dreams And Imitation In Childhood*, 1st Edition., vol. Vol. 24. London: Routledge, 1951.

[11] J. M. Gottman, «The world of coordinated play: Same- and cross-sex friendship in young children.», 1986. [En línea]. Disponible en: https://api.semanticscholar.org/CorpusID:140430088

[12] S. M. Pellis y V. C. Pellis, «Rough-and-tumble play and the development of the social brain.», *Curr Dir Psychol Sci*, vol. 16, n.º 2, pp. 95-98, 2007, doi: 10.1111/j.1467-8721.2007.00483.x.

[13] C. C. Abt, *Serious games*. University press of America, 1987.

[14] U. Ritterfeld, M. Cody, y P. Vorderer, *Serious games: Mechanisms and effects*. Routledge, 2009.

[15] F. Laamarti, M. Eid, y A. El Saddik, «An Overview of Serious Games», *International Journal of Computer Games Technology*, vol. 2014, n.º 1, p. 358152, ene. 2014, doi: https://doi.org/10.1155/2014/358152.

[16] V. Wattanasoontorn, R. J. G. Hernández, y M. Sbert, «Serious games for e-health care», *Simulations, Serious Games and Their Applications*, pp. 127-146, 2014.

[17] P. M. Kato, «Video Games in Health Care: Closing the Gap», *Review of General Psychology*, vol. 14, n.º 2, pp. 113-121, jun. 2010, doi: 10.1037/a0019441.

[18] P. Rego, P. M. Moreira, y L. P. Reis, «Serious games for rehabilitation: A survey and a classification towards a taxonomy», en *5th Iberian Conference on Information Systems and Technologies*, 2010, pp. 1-6.

[19] J. W. Burke, M. D. J. McNeill, D. K. Charles, P. J. Morrow, J. H. Crosbie, y S. M. McDonough, «Optimising engagement for stroke rehabilitation using serious games», *Vis Comput*, vol. 25, pp. 1085-1099, 2009.

[20] A. N. Krichevets, E. B. Sirotkina, I. V Yevsevicheva, y L. M. Zeldin, «Computer games as a means of movement rehabilitation», *Disabil Rehabil*, vol. 17, n.º 2, pp. 100-105, 1995.

[21] B. A. Primack *et al.*, «Role of Video Games in Improving Health-Related Outcomes: A Systematic Review», *Am J Prev Med*, vol. 42, n.º 6, pp. 630-638, 2012, doi: https://doi.org/10.1016/j.amepre.2012.02.023.

[22] P. Kato, S. Cole, D. Bradlyn, y B. Pollock, «A Video Game Improves Behavioral Outcomes in Adolescents and Young Adults With Cancer: A Randomized Trial», *Pediatrics*, vol. 122, pp. e305-17, sep. 2008, doi: 10.1542/peds.2007-3134.

[23] M. Eckert *et al.*, «The Blexer system – Adaptive full play therapeutic exergames with web-based supervision for people with motor dysfunctionalities», *EAI Endorsed Transactions on Game-Based Learning*, vol. 5, p. 155085, sep. 2018, doi: 10.4108/eai.13-7-2018.155085.

[24] J. H. Y. Loo, D. BAMIOU, N. Campbell, y L. M. Luxon, «Computer-based auditory training (CBAT): benefits for children with language-and reading-related learning difficulties», *Dev Med Child Neurol*, vol. 52, n.º 8, pp. 708-717, 2010.

[25] Sound Storm CAPD Pty Ltd, «Sound Storm».

[26] S. Cameron, H. Glyde, H. Dillon, A. King, y K. Gillies, «Results from a National Central Auditory Processing Disorder Service: A Real-World Assessment of Diagnostic Practices

and Remediation for Central Auditory Processing Disorder», *Semin Hear*, vol. 36, pp. 216-236, oct. 2015, doi: 10.1055/s-0035-1564457.

[27] L. Sampedro Jiménez, «Diseño e implementación de un videojuego terapéutico para problemas asociados con el déficit de atención», Proyecto Fin de Carrera/Grado, E.T.S.I. y Sistemas de Telecomunicación (UPM), Madrid, 2018.

[28] E. García Medina, «Memoria final de alumno en prácticas», Madrid, 2022.

[29] M. C. 'Ojeda Egocheaga, «Desarrollo e implementación de la segunda fase del videojuego terapéutico "El Planeta de Amalia"», Proyecto Fin de Carrera/Grado, E.T.S.I. y Sistemas de Telecomunicación (UPM), Madrid, 2021.

[30] B. S. 'Morar, «Ampliación de un videojuego terapéutico para mejora del procesamiento auditivo: creación de un módulo de comprensión de información.», Proyecto Fin de Carrera/Grado, E.T.S.I. y Sistemas de Telecomunicación (UPM), Madrid, 2023.

[31] M. 'Ortega García, «Ampliación de un videojuego terapéutico para la mejora del procesamiento auditivo: desarrollo de un módulo de secuencias melódicas», Proyecto Fin de Carrera/Grado, E.T.S.I. y Sistemas de Telecomunicación (UPM), Madrid, 2023.

[32] P. Lucas Olivares, «Ampliación de un videojuego terapéutico para mejora del procesamiento auditivo: creación de un módulo de escucha dicótica», Proyecto Fin de Carrera/Grado, E.T.S.I. y Sistemas de Telecomunicación (UPM), Madrid, 2023.

[33] Aurea TAV, «Centro de audiología Aurea TAV». Accedido: 28 de junio de 2024. [En línea]. Disponible en: https://www.aureatav.es/

[34] A.-M. Gabaldón-Pérez, M. Dolón-Poza, M. Eckert, N. Máximo-Bocanegra, M.-L. Martín-Ruiz, y I. Pau De La Cruz, «Serious Game for the Screening of Central Auditory Processing Disorder in School-Age Children: Development and Validation Study», *JMIR Serious Games*, vol. 11, abr. 2023.

[35] J. Blauert, *Spatial hearing: the psychophysics of human sound localization*. MIT press, 1997.

[36] B. Gardner y K. Martin, «HRFT Measurements of a KEMAR Dummy-head Microphone». 1994.

[37] O. Lauzirika Zarrabeitia, «Espacializaci´on de sonido en un entorno tridimensional», Madrid, feb. 2022.

[38] J. K. Haas, «A history of the unity game engine», 2014.

[39] «Unity Game Engine». Accedido: 28 de junio de 2024. [En línea]. Disponible en: https://unity.com/es

[40] Lenovo, «Lenovo Explorer». Accedido: 28 de junio de 2024. [En línea]. Disponible en: https://www.lenovo.com/es/es/p/smart-devices/virtual-and-augmented-reality/lenovo-explorer/lenovo-explorer/g10nreag0a2?orgRef=https%253A%252F%252Fwww.google.com%252F

[41] Microsoft, «Instale el software de Windows Mixed Reality». Accedido: 28 de junio de 2024. [En línea]. Disponible en: https://learn.microsoft.com/es-es/windows/mixed-reality/enthusiast-guide/install-windows-mixed-reality

[42] Microsoft, «¿Qué es Mixed Reality Toolkit 2?» Accedido: 28 de junio de 2024. [En línea]. Disponible en: https://learn.microsoft.com/es-es/windows/mixed-reality/mrtk-unity/mrtk2/?view=mrtkunity-2022-05#documentation

[43] A. Marcos y N. Zagalo, «Instantiating the creation process in digital art for serious games design», *Entertain Comput*, vol. 2, n.º 2, pp. 143-148, 2011, doi: https://doi.org/10.1016/j.entcom.2010.12.003.

[44] J. M. Blain, *The complete guide to Blender graphics: computer modeling & animation*. AK Peters/CRC Press, 2019.

[45] Pure Poly, «Free Low Poly Nature Forest». Accedido: 2 de julio de 2024. [En línea]. Disponible en: https://assetstore.unity.com/packages/3d/environments/landscapes/free-low-poly-nature-forest-205742

[46] Gamemag Creation Studio, «Oriental Structure Pack - Lite». Accedido: 2 de julio de 2024. [En línea]. Disponible en: https://assetstore.unity.com/packages/3d/oriental-structure-pack-lite-81021

[47] Valem Tutorials, «Oculus Hands How to Make a VR Game 2022.unitypackage». Accedido: 4 de julio de 2024. [En línea]. Disponible en: https://drive.google.com/file/d/10b39IekUdpBHlcTslZ-BlNRyH5uqPUe1/view

[48] FreeSound, «Free audio resources». Accedido: 2 de julio de 2024. [En línea]. Disponible en: https://freesound.org/

[49] GLASSDOOR, «Sueldos para el puesto de Desarrollador De Videojuegos en España». Accedido: 2 de julio de 2024. [En línea]. Disponible en: https://www.glassdoor.es/Sueldos/desarrollador-de-videojuegos-sueldo-SRCH_KO0,28.htm

[50] Naciones Unidas, «Objetivos de Desarrollo Sostenible». Accedido: 2 de julio de 2024. [En línea]. Disponible en: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/

Anexo A. : Configuración en Unity para desarrollar un proyecto de Realidad Virtual

Este capítulo trata sobre la configuración del entorno para un proyecto RV en Unity Engine, explicando primero los pasos detallados de la preparación para que un dispositivo RV puede funcionar dentro del proyecto de Unity, y luego describiendo con brevedad cómo implementar rápidamente las funciones de control del jugador e interactuar con los objetos.

Como ya se ha mencionado en el cuerpo del texto, se puede crear cualquier proyecto 3D para desarrollar juegos RV, siempre que se haya instalar XR Plug-in Management, en la figura 28, se puede ver que hay algunos errores después de instalar el plug-in, se hace clic en el signo de exclamación amarillo, puede seguir los pasos que se indica Unity, para resolver el problema, el problema en la figura es que, después de habilitar XR en el proyecto, es necesario añadir al menos un tipo de perfil de interacción, se hace clic en el botón Editar, puede ir a la pestaña que se muestra en la figura 29, se hace clic en el botón más junto a la lista del perfil de interacción para añadir el dispositivo RV correspondiente a la lista, aquí con el fin de añadir todos ellos como medida de precaución, no habrá ningún problema.

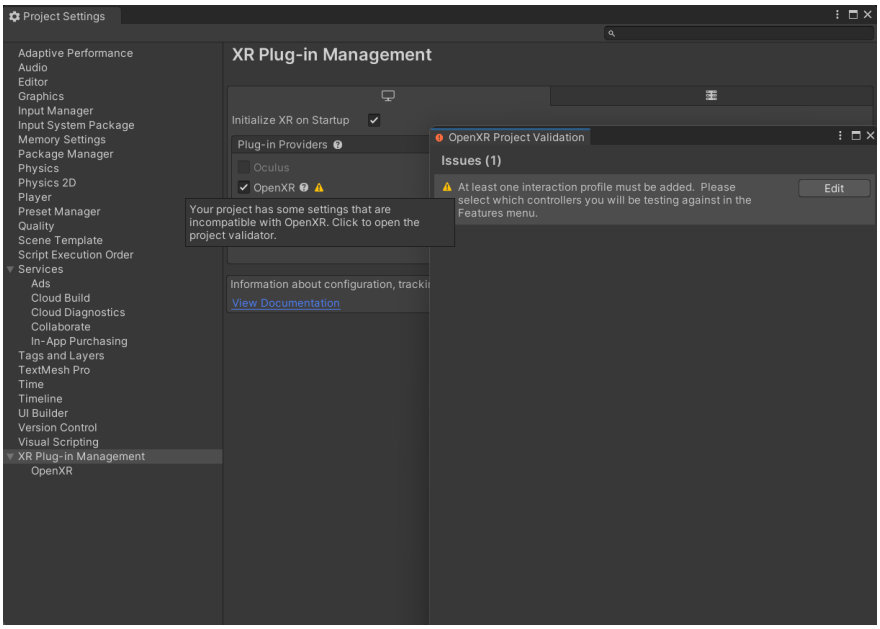


Figura 28: Resolver errores para la instalación correcta de XR Plug-in Management.

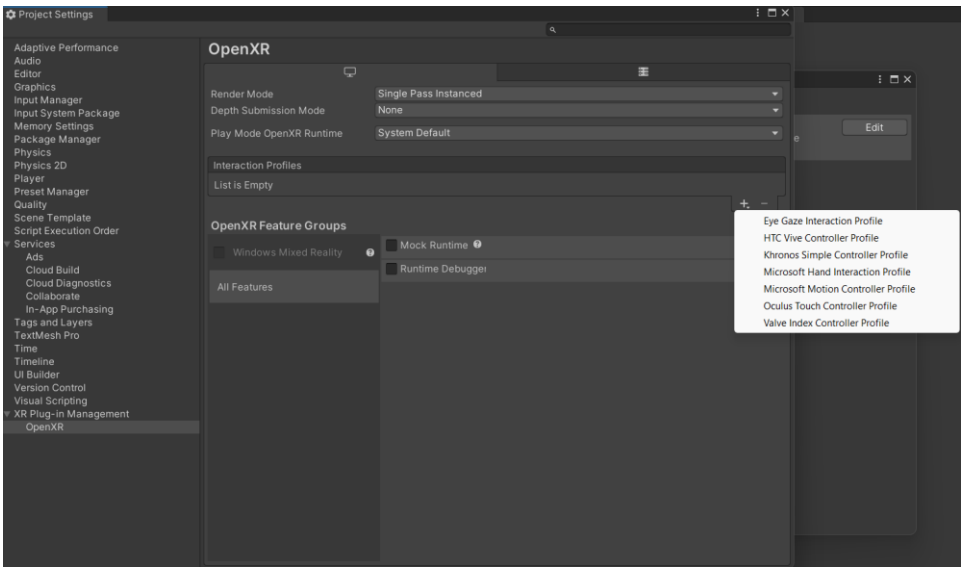


Figura 29: Añadir perfiles de interacción para resolver errores.

Después de resolver el primer problema, ya que todos los dispositivos RV que admiten la plataforma Windows Mixed Reality necesitan seleccionar la opción del grupo de características de Windows Mixed Reality en la configuración, pero en ese momento, esta opción no se puede seleccionar. El sistema indica que es necesario descargar un paquete para instalar ese plugin. Simplemente haciendo clic en el icono de interrogación al lado de la opción, se puede descargar el paquete de herramientas directamente desde el sitio oficial de Microsoft.

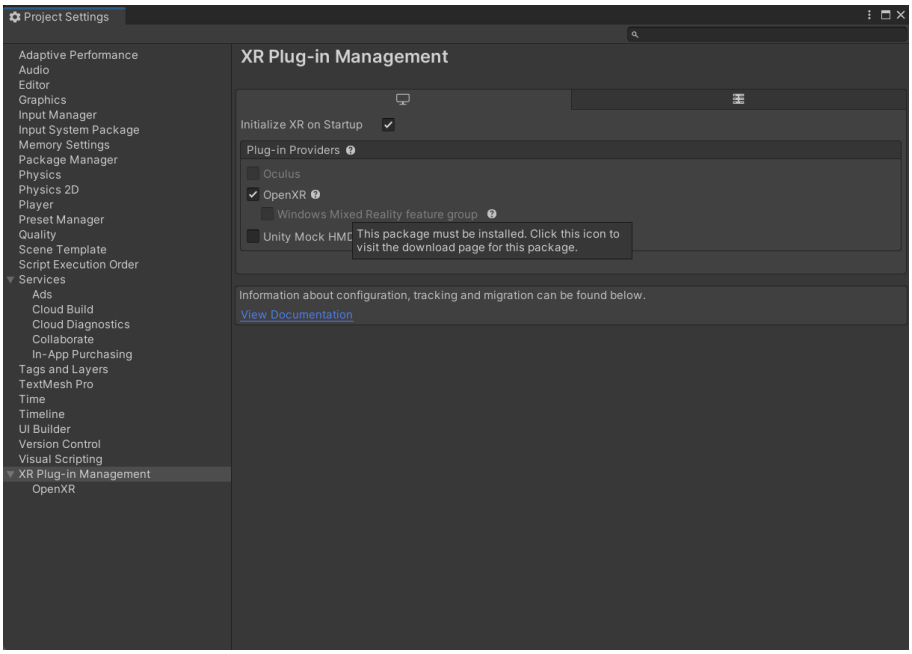


Figura 30: Icono para para abrir la página wed de decargar Mixed Reality Feature Tool for Unity.

El sitio web que se abre tiene pasos detallados para instalar el Mixed Reality Feature Tool for Unity. La figura 31 muestra algunos de los pasos importantes. Después de abrir el archivo exe para instalar, el primer paso es seleccionar la ruta del proyecto Unity donde se instalará este paquete de herramientas, y el segundo paso es elegir las funciones a instalar, entre las cuales solo el Mixed Reality OpenXR Plugin es obligatorio, mientras que otras herramientas se pueden instalar según las necesidades, como el plugin de audio espacial. Una vez completada la instalación, Unity automáticamente importará y compilará el proyecto, y después de esto, el proyecto estará listo para funcionar con dispositivos RV.

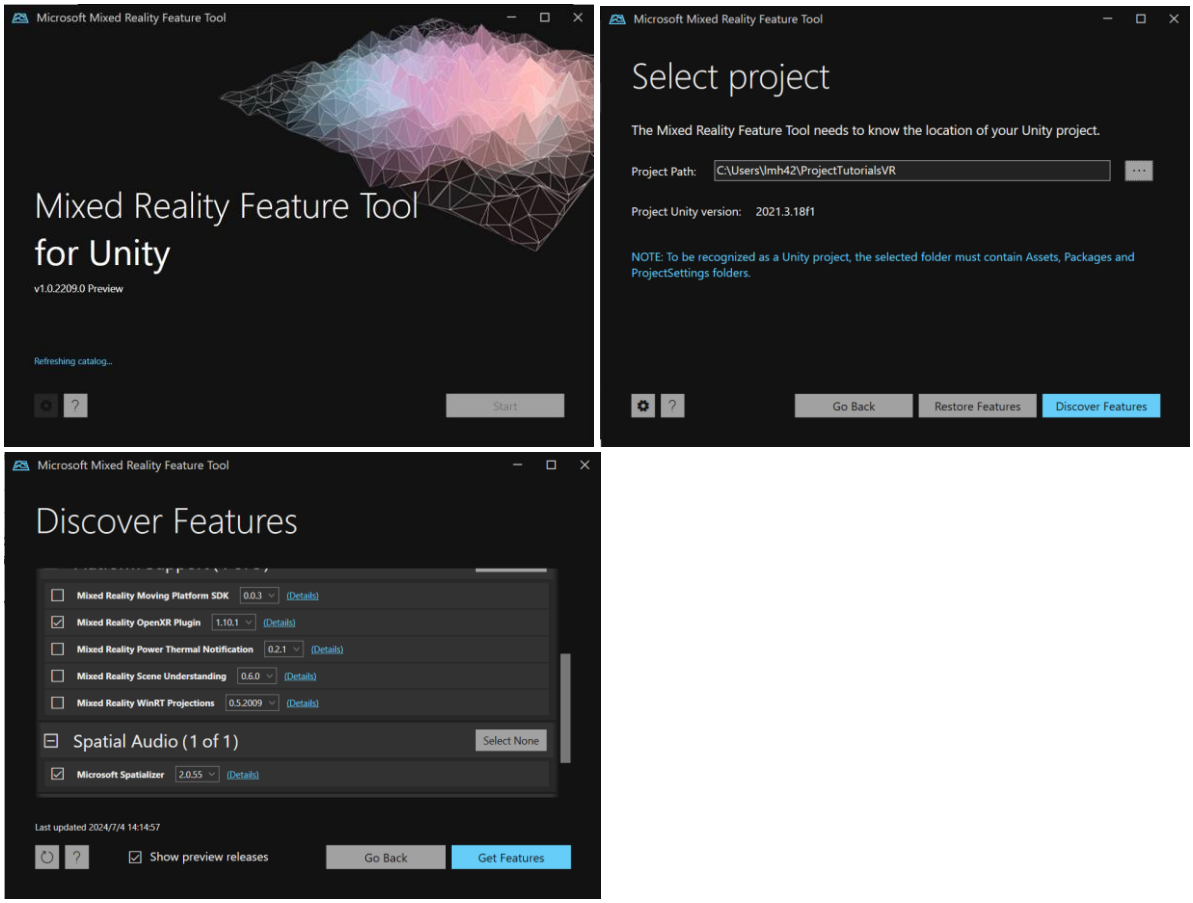


Figura 31: Pasos importantes de la instalación de Mixed Reality Feature Tool for Unity.

A continuación, se explica cómo usar los paquetes de herramientas y plugins proporcionados por Unity para realizar rápidamente la entrada de comandos en el juego utilizando los mandos RV. Primero, es necesario descargar e instalar el XR Interaction Toolkit desde el Package Manager de Unity, y luego importar los Starter Assets y XR Device Simulator incluidos en ese plugin.

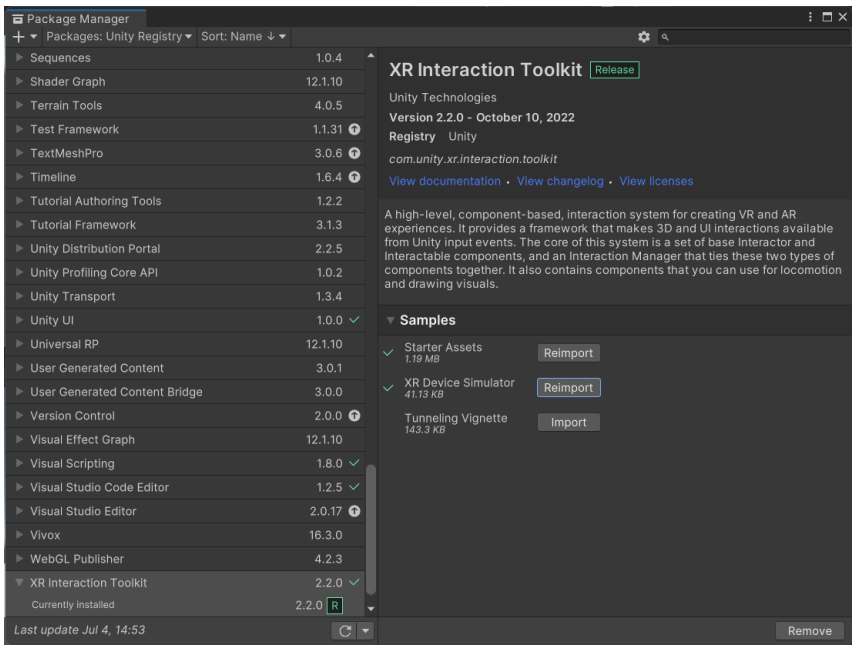


Figura 32: XR Interaction Toolkit.

La figura 33 muestra la interfaz de XRI Default Input Actions, proporcionada por el paquete de Starter Assets. Su función es definir y gestionar una serie de acciones de entrada (como agarrar y seleccionar) y mapear estas acciones a las entradas de dispositivos XR (RV, RA), como botones de controladores y touchpads. En esta interfaz de configuración, los usuarios pueden definir cualquier acción, como en la figura 33, donde la acción de entrada Select Value del XRI LeftHand Interaction está vinculada al botón de agarre del controlador izquierdo.

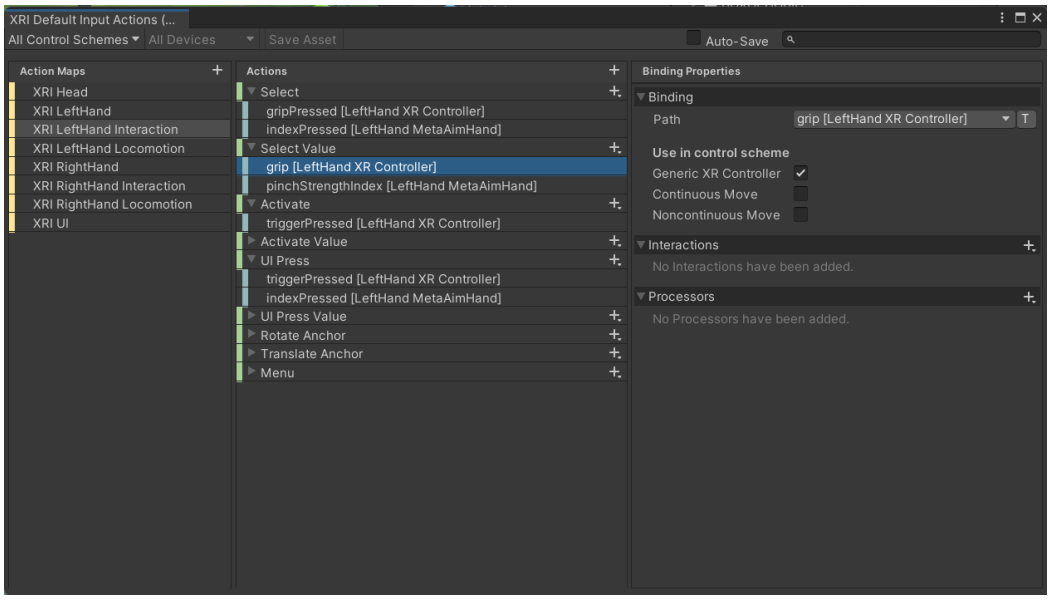


Figura 33: Pantalla de configuración de XRI Default Input Actions.

Una vez que se comprenden las XRI Default Input Actions, se pueden comenzar a configurar acciones de entrada para los mandos RV. En un proyecto RV, es necesario crear un XR Origin en la escena como el principal objeto de control del jugador (ver figura 34 izquierda), que está equipado con una cámara y objetos hijos de controladores de manos izquierda y derecha. En la figura del centro de la figura 34, se puede ver que el componente de script XR Controller del controlador tiene varias acciones. Añadiendo acciones a este componente se pueden realizar entradas mediante el mando. Los desarrolladores pueden escribir sus propios scripts de acción, pero el método más simple es usar los archivos de preset proporcionados en el paquete de XRI Default Input Actions, como se muestra en la figura más a la derecha de la figura 34, que ya tiene todas las acciones configuradas para el objeto correspondiente. Configurándolo directamente al objeto LeftHand Controller, se puede interactuar directamente en el juego usando el mando para controlar la mano izquierda.

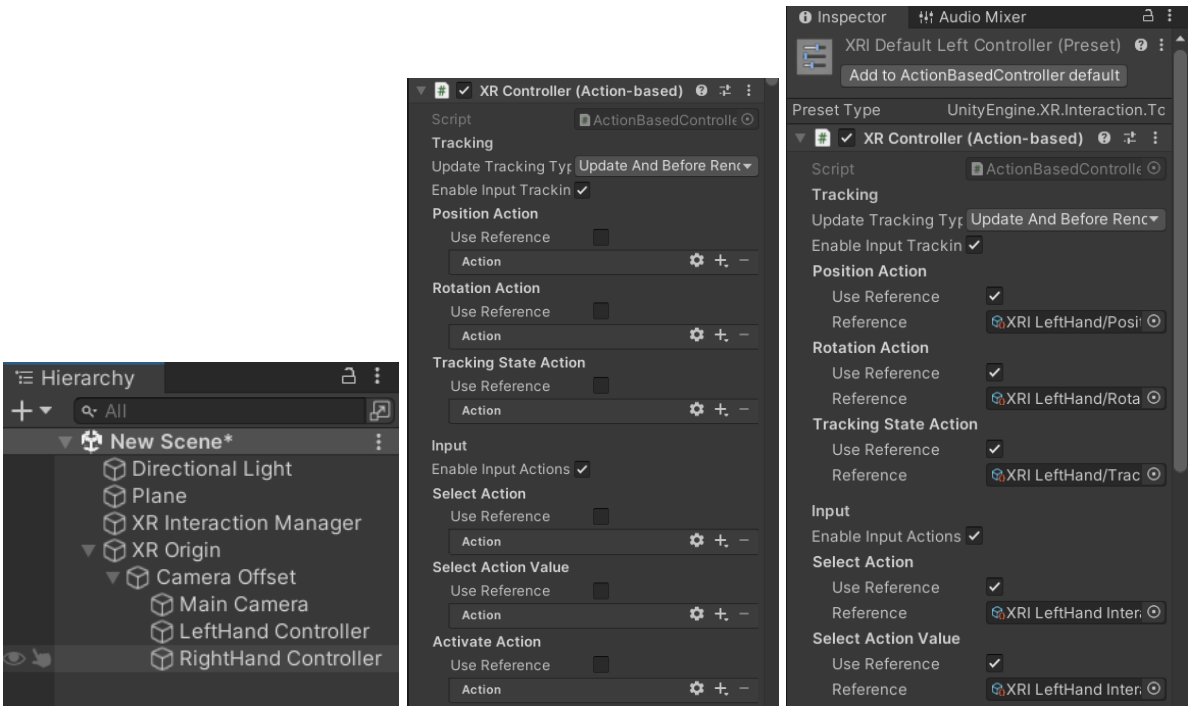


Figura 34: XRI Default action para XR Controller.

En el objeto XR Origin, además de configurar el componente XR Controller que permite que los mandos funcionen, también se necesitan una serie de componentes adicionales (figura 35), que generalmente tienen funciones preestablecidas y no requieren mucha configuración o modificación:

- **XR Origin**: Es el componente central del XR Interaction Toolkit, utilizado para gestionar las configuraciones básicas y la posición de las cámaras y controladores en la experiencia RV.
- **Input Action Manager**: Administra los activos de acciones de entrada en el sistema de entrada, manejando eventos de entrada de todos los dispositivos.
- **Locomotion System**: Administra el sistema de movimiento dentro del entorno RV.

- **Character Controller**: Este componente es común en todos los tipos de proyectos en Unity para implementar el movimiento del personaje y la detección de colisiones, y en este proyecto se utiliza para activar el juego en el área de inicio.
- **Character Controller Driver**: Coordina la interacción entre el Character Controller y el XR Origin, asegurando que el movimiento físico del jugador sea consistente con el movimiento dentro del sistema XR.
- **Continuous Turn Provider (Action-based)**: Maneja la rotación continúa basada en acciones, permitiendo que el jugador gire suavemente a través de entradas del controlador.
- **Snap Turn Provider (Action-based)**: Maneja la rotación instantánea basada en acciones, permitiendo que el jugador realice giros rápidos a ángulos fijos. Estos dos componentes de giro no son obligatorios, ya que girar la cabeza con las gafas RV también puede lograr la rotación, pero se agregan para facilitar el juego en espacios reducidos o en situaciones donde mover el cuerpo o el cuello puede ser difícil.

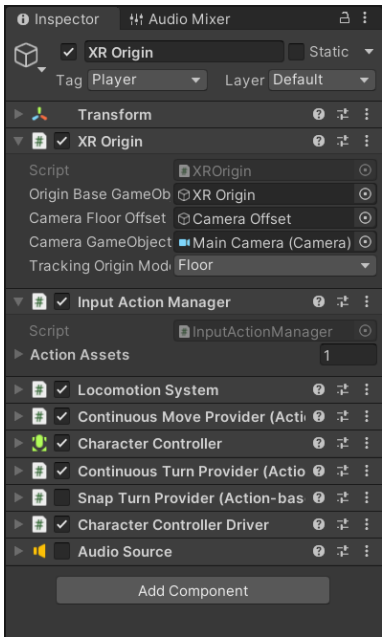


Figura 35: Las componentes añadidas en el objeto XR Origin.

Finalmente, para que la interacción sea efectiva, no solo es necesario que el juego pueda recibir entradas del mando, sino que los objetos con los que se interactúa en el juego también deben poder reaccionar adecuadamente. En este proyecto, el único objeto interactivo disponible actualmente es el farolillo volador, utilizado para la localización del sonido. Este objeto puede responder a las acciones de selección del jugador mediante la adición del componente XR Simple Interactable.

Anexo B. Manual de usuario

B.1.1. Controles del usuario

El control de "Pandalia" es muy simple, pero se divide en dos tipos. Cuando se inicia un proyecto Unity sin conectar un dispositivo de RV, el plugin XR ofrece una herramienta de simular operaciones mediante teclado y ratón. La figura 37 muestra todos los controles del juego. Los jugadores pueden moverse usando W, S, A, D, girar la cabeza mediante el ratón, y luego utilizar la tecla Tab para alternar entre moverse y controlar los mandos izquierdo y derecho, una vez alterado, el ratón no controla la cabeza, sino la mano. Es decir, los jugadores no pueden controlar las manos en el juego mientras se mueven y viceversa. Al controlar las manos, con el clic izquierdo del ratón, se selecciona farolillos.

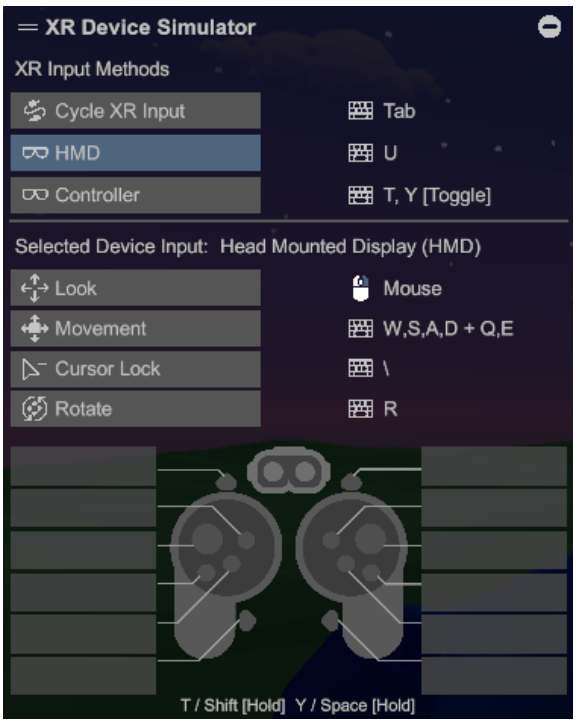


Figura 36: Controles del usuario cuando se utiliza el simulador.

Cuando se utiliza las gafas de RV con los controladores como los mostrados en la siguiente figura, esos son los mandos de las gafas de Lenovo Explorer. Esta figura presenta básicamente el control en este proyecto. En la parte superior del mando, hay un botón de control direccional (donde está el pulgar en la figura), que permite controlar el movimiento en el juego. En la posición del dedo índice en la figura, hay un botón de disparo (trigger), que se utiliza para seleccionar los farolillos en el juego.

Figura 37: Los mandos de un dispositivo RV de modelo Lenovo Explorer.

Para tener más detalles, la figura 39 presenta dos tipos más comunes de mandos de RV. A la izquierda está el controlador de dispositivo de RV VIVE de la marca HTC, y a la derecha está el controlador del dispositivo RV de Oculus. El Touchpad del primero y el Thumbstick del segundo son los botones utilizados para el control de movimiento en este juego. El botón Trigger en ambos controladores se utiliza para seleccionar el farolillo.

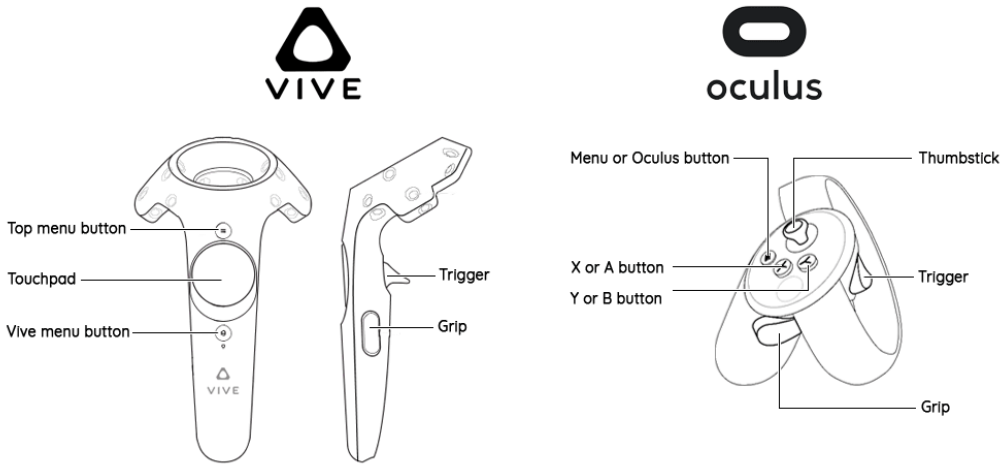


Figura 38: Diagrama del controlador de RV.

B.1.2. Guía para el usuario

Una vez que se haya comprendido el modo de control del juego, en esta sección se resumirá el proceso general del juego y se explicará cómo jugar.

Primero, a través del enlace a continuación, descargue el archivo zip en drive. Puede haber diferentes versiones, elija la versión más reciente para descargar. Después de descomprimirlo, ejecute pandalia.exe para iniciar el juego.

https://drive.google.com/drive/folders/1LdNLVCKBSYPHoqMkz76AsO4iYNJGnBei?usp=drive_link

Una vez dentro del juego, el jugador puede seguir los siguientes pasos:

1. Iniciar sesión en el juego: Después de entrar al juego, el jugador verá la pantalla de inicio de sesión como se muestra en la Figura 40. En los dos campos de entrada de la interfaz, se introduce el usuario registrado y la contraseña correspondiente. Si es un nuevo usuario, puede introducir cualquier nombre de usuario y contraseña para registrar una nueva cuenta. Si el nombre de usuario introducido ya está registrado, el sistema indicará un error de contraseña. Si el nombre de usuario no existe, el sistema preguntará si desea registrar un nuevo usuario.

Figura 39: Pantalla de inicio de sesión.

2. Configuración del juego: Después de iniciar sesión correctamente, el juego entrará a la pantalla mostrada en la Figura 41. Haciendo clic en el botón Volúmenes, se accederá a la pantalla de configuración del volumen del juego. Haciendo clic en el botón Modo de terapia, el juego se accederá a la pantalla de configuración del modo de terapia del juego.

Figura 40: Pantalla de configuraciones.

3. Configuración de volúmenes: En la interfaz mostrada en la Figura 42, se permite a los jugadores ajustar el volumen y seleccionar el ruido de fondo. Al hacer clic en el cuadro blanco de Tipo de ruido en la esquina superior izquierda, aparecerá un menú desplegable donde se pueden elegir varios tipos de ruido de fondo o seleccionar no aplicar ruido de fondo. Los sliders en la interfaz controlan diferentes volúmenes, de arriba hacia abajo son: volumen del ruido de fondo, volumen general, volumen del audio de entrenamiento de localización y volumen de varios sonidos de alerta dentro del juego. Además, hay dos botones que permiten probar los volúmenes de los últimos dos sonidos. Los jugadores pueden ajustar el volumen según sus necesidades (mover el slider completamente a la derecha es 0dB para el volumen máximo, y completamente a la izquierda es -80dB para el volumen mínimo). Finalmente, no se olvida hacer clic en el botón Guardar en la parte inferior después de realizar los ajustes para guardar la configuración.

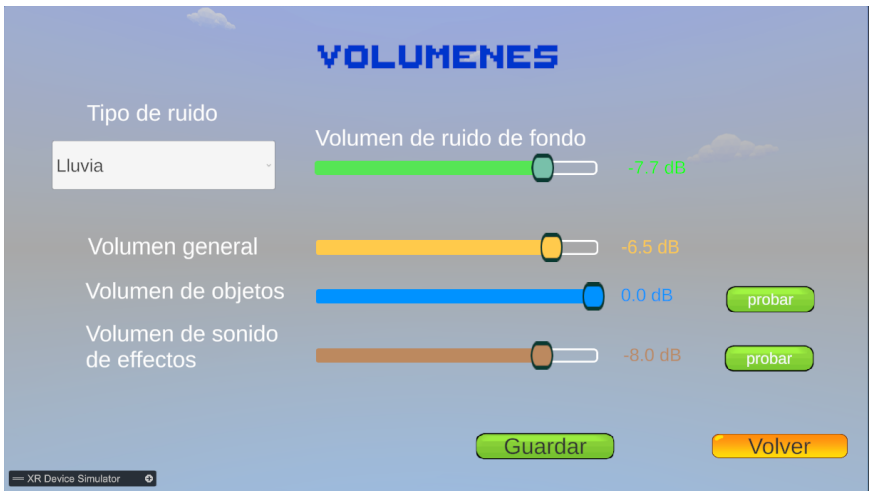


Figura 41: Configuraciones de volúmenes.

4. Configuración del juego terapéutico: En la Figura 43, los jugadores deben personalizar varios parámetros del juego según su situación de entrenamiento. Primero, hacer clic en Nivel 1 en la esquina superior izquierda de la interfaz, se abrirá un menú desplegable con cinco niveles de dificultad del 1 al 5. Cambiar a un nivel permite configurar los detalles de ese nivel. Las cuatro opciones de nivel debajo del menú desplegable determinan cuántos niveles de dificultad jugará el jugador. El Nivel 1 es obligatorio. Al seleccionar varios niveles, una vez dentro del juego, el jugador pasará al siguiente nivel después de completar las rondas de cada dificultad y cargará la configuración del próximo nivel. La función de los parámetros en esta interfaz ya se ha explicado en detalle en el texto principal. Después de guardar la configuración, hacer clic en el botón Jugar para entrar al juego.

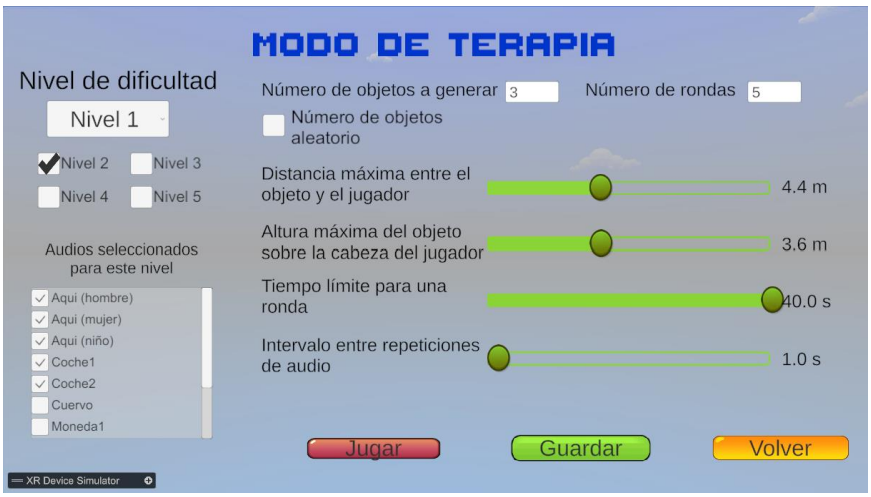


Figura 42: Configuraciones de modo de terapia.

5. Comenzar el juego: Después de entrar al juego, se visualiza la pantalla mostrada en la Figura 44. En el centro de la pantalla es el área para comenzar el juego. Utilizando los controles mencionados en el capítulo anterior (WASD en caso de utilizar simulador en ordenador, Thumbstick en caso de utilizar gafas RV y mandos), controla el personaje para que entre en el pilar de luz azul y comience el juego.

Figura 43: La escena del juego al entrar.

6. Durante el juego: Después de comenzar el juego, según la configuración del jugador, se generará un número de farolillos en posiciones aleatorias dentro de un cierto rango alrededor del jugador, y solo uno de ellos emitirá un sonido de manera continua. Cada vez que el jugador selecciona un farolillo sonoro, se considera que ha completado una ronda. Después de completar todas las rondas de todas las dificultades, el juego termina y se guarda el resultado.

7. Método para seleccionar el farolillo: Si se juega en un ordenador, primero se mueve el ratón para girar la vista de la cabeza y se busca el farolillo que emite sonido. Una vez encontrado, se presiona Tab para cambiar al control de la mano. Se controla la dirección del rayo de la mano, se mueve el rayo hacia el farolillo y se hace clic con el botón izquierdo del

ratón para seleccionarlo. Si se utilizan mandos, se gira la cabeza para buscar la fuente del sonido, se utiliza el mando para apuntar al farolillo y se presiona el trigger para seleccionarlo.

Figura 44: Pantalla durante el juego.

7. Salida de resultados: Por último, tras finalizar el juego, el usuario puede encontrar el archivo de resultados en la siguiente dirección.

C:\Users\" nombre usuario Windows" \AppData\LocalLow\DefaultCompany\pandalia

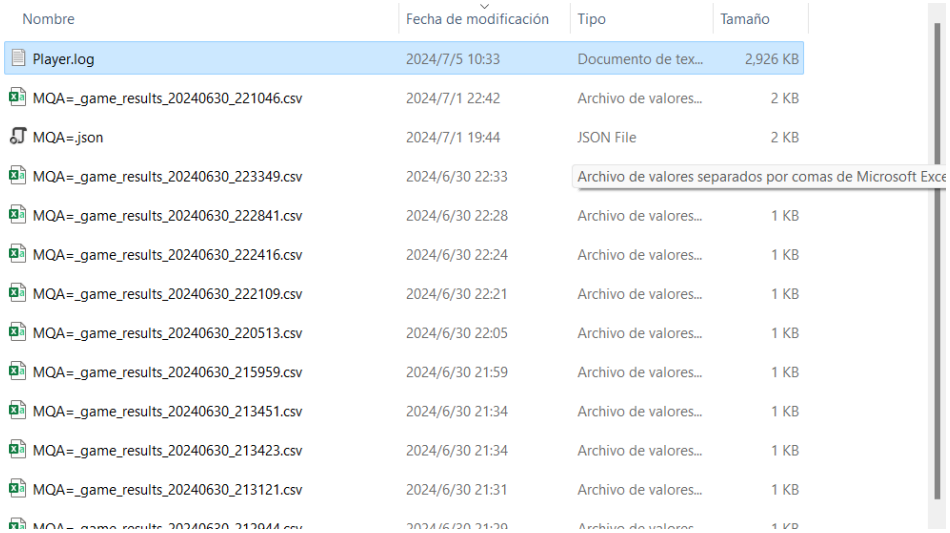


Figura 45: Archivos de resultados del juego.