

Todo list

Añadir conclusiones evaluación al resumen en español	III
Añadir conclusiones evaluación al resumen en inglés	V
Hablar sobre git	53
Cuestionarios y pruebas. Conclusiones	119
Conclusiones y líneas futuras	147
Reescribir preguntas formularios. Rellenar respuestas cuestionarios	165



ugr

Universidad
de Granada

TRABAJO FIN DE GRADO
INGENIERÍA INFORMÁTICA

Realidad Virtual en Entrenamiento Cognitivo

**Estudio de aplicación de un juego serio en realidad virtual
para tratar el deterioro cognitivo leve**

Autor
Antonio Jiménez Amador

Directores
Francisco Luis Gutierrez Vela
Patricia Paderewski Rodriguez



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

—
Granada, Septiembre de 2023

REALIDAD VIRTUAL EN ENTRENAMIENTO COGNITIVO

Autor: Antonio Jiménez Amador

Tutores: Francisco Luis Gutierrez Vela, Patricia Paderewski Rodriguez

Departamento: Lenguajes y Sistemas Informáticos

Titulación: Grado en Ingeniería Informática

Palabras clave: Realidad virtual, entrenamiento cognitivo, deterioro cognitivo, memoria, percepción, razonamiento, mayores, videojuegos, juego serio, Unity, Oculus.

Resumen

Este proyecto se ha desarrollado con el objetivo de proponer y analizar el funcionamiento de la tecnología de realidad virtual como herramienta para el tratamiento del deterioro cognitivo leve así como el entrenamiento de las habilidades cognitivas básicas como la memoria, percepción y el razonamiento. Con dicho objetivo se ha estudiado la cognición humana y la realidad virtual, para desarrollar una demostración de juego serio que pueda ser utilizado como apoyo a los tratamientos de rehabilitación cognitiva en personas que la hayan visto deteriorada, o como entrenamiento para mantener una cognición sana especialmente en etapas de edad avanzada. Se intenta que la realidad virtual sirva de aliciente y motivación para las personas a la vez que se ofrece la posibilidad de mejorar la eficacia de los tratamientos actuales. Tras el desarrollo del proyecto y posterior prueba con personas, se evalúa la usabilidad del sistema.

Añadir conclusiones evaluación al resumen en español

VIRTUAL REALITY IN BRAIN TRAINING

Author: Antonio Jiménez Amador

Supervisors: Francisco Luis Gutierrez Vela, Patricia Paderewski Rodriguez

Department: Lenguajes y Sistemas Informáticos

Degree: Grado en Ingeniería Informática

Keywords: Virtual reality, brain training, mild cognitive impairment, memory, perception, reasoning, elderly, videogames, serious game, Unity, Oculus.

Abstract

This project has been developed with the aim of proposing and analysing the use of virtual reality technology as a tool in the treatment of mild cognitive impairment as well as the training of basic cognitive skills such as memory, perception and reasoning. With this aim, human cognition and virtual reality have been studied in order to develop a demo of a serious game able to be used as support for cognitive rehabilitation treatments on people who have seen their cognition deteriorated, or as regular training in order to keep a healthy one specially at older ages. It is the intention that virtual reality aid at keeping the motivation for the people as well as allowing the possibility to improve the efficacy of current treatments. After the development of this project and later test with people, the system's usability is evaluated.

Añadir conclusiones evaluación al resumen en inglés

Yo, **Antonio Jiménez Amador**, alumno de la titulación de Grado en Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 75574393G, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Antonio Jiménez Amador

Granada a 1 de Septiembre de 2023.

D. Francisco Luis Gutierrez Vela, profesor del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Granada.

Dña. Patricia Paderewski Rodriguez, profesora del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Realidad Virtual en Entrenamiento Cognitivo, Estudio de aplicación de un juego serio en realidad virtual para tratar el deterioro cognitivo leve***, ha sido realizado bajo su supervisión por **Antonio Jiménez Amador**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 1 de Septiembre de 2023.

Los directores:

Francisco Luis Gutierrez Vela Patricia Paderewski Rodriguez

A mi familia.
Por apoyarme durante
este largo camino.

Antonio Jiménez Amador

Acrónimos

ETSITT	Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
GDD	Game Design Document
HMD	Head Mounted Display
JSON	JavaScript Object Notation
RV	Realidad Virtual
SDK	Software Development Kit
SUS	System Usability Scale
TFG	Trabajo Fin de Grado
UGR	Universidad de Granada
UI	User Interface
VR	Virtual Reality
VRTK	Virtual Reality Toolkit

Índice

Resumen	III
Abstract	v
Acrónimos	xiii
I Introducción	1
1. Introducción	3
1.1. Introducción	3
1.2. Motivación	4
1.3. Objetivo	5
1.3.1. Objetivos específicos	5
1.4. Estructura de la memoria	6
II Estudio del problema	7
2. Estado del arte	9
2.1. Cognición	10
2.2. Entrenamiento cognitivo	12
2.3. Ejercicios de Brain Training	13
2.3.1. Cuadernos de ejercicios	13
2.3.2. Juegos grupales	14

2.3.3. Juegos y plataformas online	15
2.4. Realidad Virtual	18
2.4.1. Renacimiento	19
2.4.2. Siglo XIX	21
2.4.3. Sensorama y Headsight	22
2.4.4. Espada de Damocles	23
2.4.5. Realidad virtual	24
2.4.6. Virtual Boy	25
2.4.7. Siglo XXI	26
2.4.8. Actualidad	28
2.5. Motores de videojuegos	31
2.5.1. Unity	31
2.5.2. Unreal Engine	32
2.5.3. Game Maker	34
2.5.4. Godot	35
2.6. Conclusión y justificación del proyecto	36
3. Análisis inicial del problema	39
3.1. Pruebas	40
3.1.1. Motricidad	40
3.1.2. Memoria	42
3.1.3. Lenguaje	43
3.1.4. Razonamiento	44
3.1.5. Comprensión espacial	45
3.2. Desarrollo	46
4. Tecnología a usar	49
4.1. Tecnología a usar	49
5. Metodología a usar	53
5.1. Desarrollo y Metodología Ágil	53

5.2. Game Design Document	55
5.3. Evaluación	56
6. Plan de entregas	59
6.1. Entregas	59
 III Desarrollo	 61
7. Desarrollo del proyecto	63
7.1. Entrega 1	64
7.1.1. Objetivo	64
7.1.2. Iteración 1	64
7.1.3. Iteración 2	67
7.1.4. Conclusiones de la entrega	71
7.2. Entrega 2	73
7.2.1. Objetivo	73
7.2.2. Iteración 1	73
7.2.3. Iteración 2	79
7.2.4. Iteración 3	87
7.2.5. Conclusiones de la entrega	90
7.3. Entrega 3	91
7.3.1. Objetivo	91
7.3.2. Iteración 1	91
7.3.3. Iteración 2	97
7.3.4. Iteración 3	103
7.3.5. Conclusiones de la entrega	104
7.4. Entrega 4	105
7.4.1. Objetivo	105
7.4.2. Iteración 1	105
7.4.3. Iteración 2	110

7.4.4. Iteración 3	118
7.4.5. Conclusiones de la entrega	119
7.5. Entrega 5	119
7.5.1. Objetivo	119
7.5.2. Iteración 1	119
7.5.3. Iteración 2	143
7.5.4. Conclusiones de la entrega 5	143
IV Conclusiones	145
Conclusiones y líneas futuras	147
V Bibliografía	149
Bibliografía	156
VI Apéndices	157
A. Game Desing Document	159
A.1. Información general	159
A.1.1. Resumen del juego	159
A.1.2. Objetivos a alcanzar por el juego	160
A.1.3. Justificación del juego	160
A.1.4. Core gameplay	160
A.1.5. Características del juego	160
A.1.6. Características del jugador	161
A.1.7. Recursos iniciales	161
A.2. Mecánicas	162
A.2.1. Elementos de juego	162
A.2.2. Reglas	162

A.2.3. Elementos de juego: Mundo	162
A.2.4. Elementos de registro y progreso	162
A.3. Dinámica	163
B. Cuestionarios	165
B.1. Plantilla	165
B.2. Resultados usabilidad	167

Índice de figuras

2.1.	Gráfica que representa la curva creada por el efecto se posición serial. [1]	11
2.2.	Cuaderno Esteve. [2]	14
2.3.	Cuaderno Rubio. [3]	14
2.4.	Juego de grupo con pelota. [4]	15
2.5.	Algunos de los juegos disponibles en Lumosity.	16
2.6.	Demostración de NeuronUp multidispositivo. [5]	17
2.7.	Juego de cálculo en Koji's Quest. [6]	17
2.8.	Demostración de Enhance. [7]	18
2.9.	Roman de la Rose. [8]	20
2.10.	La última Cena. Leonardo Da Vinci. [9]	20
2.11.	Entrega de las llaves a San Pedro. Pietro Perugino. [10]	20
2.12.	Panorama de Londres. [11]	21
2.13.	Sección de la Rotunda en Leicester Square. [12]	21
2.14.	Estereoscopio de David Brewster. [13]	22
2.15.	Figura 5 de la patente del Sensorama. [14]	23
2.16.	Headsight de Philco Corporation. [15]	23
2.17.	Espada de Damocles de Ivan Sutherland y Bob Sproull. [16]	24
2.18.	Eyephone y Dataglove junto al sistema de seguimiento Polhemus en exposición Nissho Iwai en Tokio en 1999. [17]	25
2.19.	Virtual Boy de Nintendo. [18]	26
2.20.	Mario's Tennis. Videojuego de Nintendo para Virtual Boy. [19]	26
2.21.	Primer kit de desarrollo de Oculus Rift. [20]	27

2.22. PlayStation VR. [21]	27
2.23. Google Cardboard ensamblada. [22]	28
2.24. HTC Vive y todos sus complementos. [23]	28
2.25. Kit de visor Valve Index junto a controles y estaciones base. [24] . .	29
2.26. Meta Quest 2. [25]	30
2.27. Interfaz de Unity. [26]	31
2.28. Monument Valley 2. [27]	32
2.29. Interfaz de Unreal Engine 4. [28]	33
2.30. Visión del jugador durante una partida de Fortnite. [29]	33
2.31. Interfaz de GameMaker Studio 2. [30]	34
2.32. Captura de pantalla de Crashlands. [31]	35
2.33. Interfaz de Godot en modo web. [32]	36
2.34. Captura de pantalla de RivenTails: Defense. [33]	36
 3.1. Plató del programa de televisión 'Atrapa un millón'. [34]	40
3.2. Boceto del escenario para la prueba de baile.	41
3.3. Boceto de la prueba de turismo. Se presenta una imagen de la torre Eiffel con decorado extra al frente.	43
3.4. Boceto para la prueba de asociación de objetos, mostrando dos zonas (verde y azul) para la clasificación de los objetos.	45
3.5. Boceto de la vista cenital del escenario principal, mostrando una posible colocación de la fuente de sonido para la prueba de localización de sonidos.	47
 4.1. Meta Quest 2. [35]	50
4.2. Interfaz de Unity. [26]	51
4.3. Portada de la Asset Store.	52
 5.1. Diagrama de desarrollo iterativo. [36]	54
5.2. Diagrama de desarrollo evolutivo. [37]	54
5.3. Representación de los resultados de un SUS. [38]	57
 7.1. Importación del paquete SteamVR Unity Plugin 2.6.0b1.	65

7.2. Líneas a añadir en manifest.json, siendo X.Y.Z la versión deseada.	66
7.3. Errores al importar VRTK Prefabs.	66
7.4. Ventana SteamVR Input.	67
7.5. Avatar por defecto de SteamVR.	68
7.6. Script 'Linked Alias Association Collection' enlazado con los objetos que representan las manos y la cabeza.	69
7.7. Script para obtener las velocidades de un objeto de SteamVR.	69
7.8. TrackedAlias enlazado con el objeto Player.	69
7.9. Script para el manejo de acciones.	70
7.10. Manejadores para la acción de agarrar, activada por el gatillo del mando.	70
7.11. Interactors anidados en el objeto TrackedAlias.	71
7.12. Interactor Facade con el manejador de acción definido anteriormente.	71
7.13. Objeto interactivo con los componentes necesarios para la RV.	72
7.14. Avatar cogiendo una esfera interactiva.	72
7.15. Ejemplo del atributo Transform en Unity que representa la posición de un objeto en el espacio.	74
7.16. Boceto que muestra el uso de un trigger (cubo verde) para detectar el controlador en su interior.	74
7.17. Boceto que representa una posible prueba de figuras. En azul el modelo a seguir, en verde los trigger necesarios.	75
7.18. Boceto representando una prueba de baile o movimientos. En gris la referencia a seguir, en verde los trigger y los movimientos que siguen.	76
7.19. Boceto de una prueba de objetivos. Los objetivos en azul y rojo deberán tocarse con el mando del mismo color.	76
7.20. Boceto de la prueba de agrupación de objetos. Las categorías "frío/z comida" no son visibles para el jugador.	77
7.21. Fotografía de un ejercicio de figuras superpuestas en formato de papel.	78
7.22. Boceto que representa la prueba de asociación de un sonido con el objeto que lo produce.	79

7.23. Variables en el script para los triggers.	80
7.24. Métodos para manejar la entrada y salida de objetos en un trigger.	81
7.25. Parte del script que controla el número de mandos dentro de un trigger.	82
7.26. Representación de la matriz con 3 filas y 4 columnas de triggers. En el centro, una capsula verde representa al jugador, y los cubos blancos sus manos.	82
7.27. Ejemplo de la estructura JSON representando un nivel con 3 partes.	83
7.28. Ejemplo que muestra dos plataformas verdes con dos InteractableSnapZone. En rosa se remarca la posición que ocupará el cubo verde si se suelta dentro de una de dichas zonas.	85
7.29. Dos métodos del script que controla los objetos dentro de cada snap zone.	86
7.30. Jugador pulsando un botón.	86
7.31. Muestra de la configuración de una fuente de audio en Unity.	88
7.32. Ventana de configuración en la que es necesaria seleccionar OculusSpatializer para utilizar el sonido 3D.	88
7.33. Método que comprueba si la fuente de audio es visible por el jugador. Este script debe estar en la propia fuente.	89
7.34. Mesa con objetos de prueba para las interacciones básicas, snap zones y botones.	89
7.35. Plató del programa de televisión 'Atrapa un millón'. [34]	92
7.36. Plató del programa de televisión '¡Ahora caigo!'. [39]	93
7.37. Boceto de la vista superior del escenario principal. En azul la zona de público, en verde la zona de atrezo, en negro la parte intercambiable para las pruebas.	93
7.38. Boceto de la vista de perfil del escenario principal. En azul la zona de público, en verde la zona de atrezo, en negro la parte intercambiable para las pruebas.	94
7.39. Boceto del escenario para la prueba de baile.	95
7.40. Boceto de la decoración para la prueba de turismo. En la pantalla una imagen de la torre Eiffel, con decorado de viajes: avión, globo terráqueo y globo aerostático.	95
7.41. Boceto de la decoración para la prueba de adivinar la canción.	96

7.42. Boceto para la prueba de figuras, con una silueta acercándose al jugador.	97
7.43. Boceto con la mesa para la prueba de asociación de sonidos. Presenta distintos objetos con botones que corresponde a cada uno.	98
7.44. Boceto para la prueba de asociación de objetos, mostrando dos zonas (verde y azul) para la clasificación de los objetos.	98
7.45. Boceto de la vista cenital del escenario principal, mostrando una posible colocación de la fuente de sonido para la prueba de localización de sonidos.	99
7.46. Parte central del escenario.	99
7.47. Plataforma principal con el escenario de baile y las paredes que rodean todo el perímetro.	100
7.48. Detalle de la colocación de la pantalla, el pedestal secundario y el hueco del ascensor.	100
7.49. Ascensor simple.	101
7.50. Ascensor con mesa.	101
7.51. Ascensor con mesa extendida.	101
7.52. Vista del escenario de baile completo.	101
7.53. Modelos para los objetivos que el jugador debe golpear.	102
7.54. Presentación del decorado para la prueba de turismo.	102
7.55. Mesa con decorado para la prueba de adivinar la canción.	103
7.56. Presentación de las figuras que actúan como público.	103
7.57. Montaje final del escenario para la entrega 3.	104
7.58. Torre Eiffel, París, Francia.	108
7.59. Coliseo romano, Roma, Italia.	108
7.60. Estatua de la libertad, Nueva York, EE. UU.	108
7.61. Médico vendando una mano.	109
7.62. Madre preparando a su hijo para el colegio.	109
7.63. Gente bailando en la feria de abril.	109
7.64. Estructura de la clase 'Prueba'.	112
7.65. Sobrescritura del método 'CargarPrueba' para la prueba de asociación de un sonido con un objeto.	113

7.66. Matriz de señales que indican al usuario dónde colocar sus manos.	114
7.67. Audio Source de Unity mostrando el clip actual que reproduce, así como otros de sus parámetros.	115
7.68. Parámetros de un objeto plano de Unity mostrando una imagen del coliseo romano.	116
7.69. Ejemplo de prueba de agrupación de objetos.	117
7.70. Estructura de objeto para la mesa.	117
7.71. Ejemplo de prueba de asociación de sonidos.	117
7.72. Código que asigna la llamada al método para terminar la prueba de forma correcta al botón adecuado.	118
7.73. Aviso sobre paquetes obsoletos.	120
7.74. Jerarquía del objeto OculusInteractionSampleRig incluido con el SDK de Meta.	121
7.75. Manos virtuales captadas usando el seguimiento de Meta Quest 2.	122
7.76. <i>Interactors</i> que permiten coger, pulsar o lanzar rayos.	123
7.77. Zona de detección formada por un <i>trigger</i> sobre una plataforma.	124
7.78. Código de la clase DetectZone que gestiona cada zona de detección.	125
7.79. Diagrama de la clase GameManager.	127
7.80. Grafo dirigido de los posibles cambios de estado durante el juego.	128
7.81. Código de la clase UIManager. Método para terminar el tutorial pulsando el botón de 'Entendido' y método para cargar los botones de elección de pruebas.	130
7.82. Diagrama de clase para UIManager.	131
7.83. Diagrama de clase para PruebasManager.	132
7.84. Diagrama de clase para EscenarioManager.	133
7.85. Diagrama de clase para TriggerManager.	134
7.86. Diagrama de clases de DetectManager y DetectZone.	135
7.87. Métodos de la clase DetectZone para comprobar si los objetos que entran y salen son de la categoría adecuada.	136
7.88. Método de la clase DetectManager para comprobar si las zonas de detección contienen los objetos correctos.	137
7.89. Método asíncrono que baja el ascensor en cada fotograma.	137

7.90. Diagrama de la clase Prueba y sus clases herederas.	138
7.91. Método sobrescrito para comprobar si la prueba de asociaciones es correcta.	138
7.92. Diagrama de las clases desarrolladas para este proyecto.	140
7.93. Diagrama de secuencia de una partida del juego.	142

Índice de Tablas

B.1. Resultados de usabilidad obtenidos tras la prueba 167

Parte I

Introducción

Capítulo 1

Introducción

1.1. Introducción

El ser humano está en constante desarrollo desde que nace. Este proceso no es únicamente físico y biológico, si no que da forma a cada persona desde temprana edad mediante cambios psicológicos y cognitivos. La cognición es la capacidad de percibir información del entorno para posteriormente procesarla y obtener conocimientos y valoraciones. Esto engloba procesos cognitivos como el razonamiento, la memoria, la atención, la capacidad de resolución de problemas y los sentimientos.

Durante la vida de una persona, la cognición se desarrolla rápidamente durante sus primeros años de vida y, en condiciones normales, se mantiene estable durante la vida adulta. Ciertos motivos externos pueden causar un deterioro anormal de las capacidades cognitivas de un individuo, como son por ejemplo algunas enfermedades neurodegenerativas como el Alzheimer u otros tipos de demencia. Incluso en personas sanas, existe un deterioro natural de las habilidades cognitivas que va ligado a la edad, por lo que es común que las personas de la tercera edad sufran de problemas cognitivos como pérdidas de memoria, dificultades de comunicación o faltas de atención. Estos son problemas que las personas mayores sufren en su día a día y que en ocasiones no les permite disfrutar plenamente de su vida y ocio diario.

Afortunadamente, la cognición es una característica humana y como muchas otras, puede ser entrenada, reforzada e incluso recuperada después de verse deteriorada. Para esto existen los llamados entrenamientos cognitivos, que suelen venir asociados a ejercicios mentales. Cualquier persona puede realizar estos ejercicios y ver reforzadas sus capacidades cognitivas, pero son especialmente

útiles para prevenir un deterioro anticipado de la cognición en personas mayores y como método de rehabilitación para personas que a causa de alguna enfermedad han visto sus capacidades cognitivas dañadas.

Estos entrenamientos existen desde hace décadas, pero han ido evolucionando con el avance de las nuevas tecnologías. Ahora en vez de tratarse de ejercicios en papel que deben realizarse de manera supervisada, se puede acceder a ellos mediante aplicaciones móviles o páginas web, que permiten un uso más cómodo y atractivo.

Sin embargo, esto tiene un problema asociado: y es que, aunque los ejercicios hayan evolucionado con la tecnología, los principales usuarios de estos ejercicios cognitivos son personas mayores que suelen no estar familiarizados en absoluto con dichas tecnologías. Por tanto, esto genera una barrera adicional para su uso ya que la mayoría de las personas mayores no sabe cómo utilizar estas aplicaciones o páginas web.

Para solventar este problema es necesario facilitar al máximo su uso para los mayores, pero es aún más importante, si cabe, que las personas estén motivadas a realizar los ejercicios. Una falta de motivación puede llegar a reducir en gran medida la efectividad de los entrenamientos cognitivos.

1.2. Motivación

Hoy en día existen tecnologías como la realidad virtual o RV, que mediante el uso de un visor especial permiten sumergir al usuario en un mundo virtual con un grado de inmersión que ninguna otra tecnología puede alcanzar. Incluso para personas jóvenes y con experiencia tecnológica, la realidad virtual provoca sensaciones extraordinarias y permite vivir experiencias imposibles en la vida real. La RV es una herramienta que usada de forma adecuada puede incrementar de manera considerable la motivación de una persona, especialmente una persona mayor, ya que le abre las puertas a un mundo virtual con el que nunca ha tenido contacto, y el factor sorpresa puede motivarlo a continuar avanzando y descubriendo cosas nuevas.

Es por este motivo, que este proyecto trata de demostrar un posible uso de la realidad virtual aplicada a la realización de ejercicios cognitivos enfocados a personas mayores. Se busca resolver los dos problemas planteados: la falta de motivación y la desconexión tecnológica de las personas mayores.

1.3. Objetivo

El objetivo principal de este proyecto es crear un prototipo de un videojuego que pueda utilizarse para comprobar la efectividad del uso de la realidad virtual como herramienta para el entrenamiento cognitivo en personas mayores.

Para alcanzar este objetivo se va a desarrollar un juego serio¹, dentro del cual se presentarán al jugador distintos ejercicios cognitivos basados en ejercicios ya existentes y adaptados al uso de la RV.

1.3.1. Objetivos específicos

Para considerar cumplido el objetivo principal de este proyecto, es necesario cumplir a su vez varios subobjetivos que permitirán el desarrollo del videojuego de forma exitosa:

- Creación de un entorno de realidad virtual especialmente diseñado para ser utilizado por personas mayores.
- Evaluación de dicho entorno para comprobar que una persona mayor y sin conocimientos tecnológicos sea capaz de interactuar con él de manera cómoda.
- Diseñar ejercicios de entrenamiento cognitivos adaptados a la realidad virtual a partir de ejercicios tradicionales, manteniendo su funcionamiento central inalterado, asegurando que son efectivos.
- Diseñar ejercicios cognitivos completamente nuevos, dando pleno uso a las capacidades de la RV de las que carecen los métodos tradicionales.
- Obtener un videojuego final que sea atractivo y motivador para las personas mayores.
- El videojuego podrá ser usado de forma personal, o como herramienta guiada por profesionales.
- Reducir al mínimo las dificultades e inconvenientes que conlleva la RV, especialmente a nivel físico: incomodidades con el casco, posibles cables que limiten el movimiento del usuario, etc.

¹Juego que tiene un propósito principal no recreativo.

1.4. Estructura de la memoria

En primer lugar, se va a estudiar la cognición humana y cómo esta puede ser entrenada y desarrollada. Para ello se utilizan diferentes tipos de ejercicios cognitivos, que han ido cambiando con la evolución de las tecnologías.

Este proyecto busca desarrollar un videojuego de realidad virtual para continuar adaptando estos ejercicios a nuevas tecnologías. Por ello se va a hablar sobre la realidad virtual, su historia y la importancia que tiene hoy en día. Posteriormente se estudiarán cuatro motores de videojuegos, que son los conjuntos de software y tecnologías utilizadas para la creación de videojuegos.

Una vez estudiados estos conceptos fundamentales para el proyecto, se detallará el conjunto de herramientas software y hardware que serán utilizadas para el desarrollo del videojuego, así como la metodología que se seguirá durante el progreso de este TFG.

A continuación, se presenta un diario de desarrollo del videojuego en el que se cuenta paso a paso, y con el apoyo de figuras, todo lo relacionado al diseño y creación del videojuego para la consecución del objetivo del proyecto. Finalmente se cierra esta memoria con unas conclusiones, pensamientos finales y posibles líneas futuras que puede seguir el proyecto.

Parte II

Estudio del problema

Capítulo 2

Estado del arte

Contenido

2.1. Cognición	10
2.2. Entrenamiento cognitivo	12
2.3. Ejercicios de Brain Training	13
2.3.1. Cuadernos de ejercicios	13
2.3.2. Juegos grupales	14
2.3.3. Juegos y plataformas online	15
2.4. Realidad Virtual	18
2.4.1. Renacimiento	19
2.4.2. Siglo XIX	21
2.4.3. Sensorama y Headsight	22
2.4.4. Espada de Damocles	23
2.4.5. Realidad virtual	24
2.4.6. Virtual Boy	25
2.4.7. Siglo XXI	26
2.4.8. Actualidad	28
2.5. Motores de videojuegos	31
2.5.1. Unity	31
2.5.2. Unreal Engine	32
2.5.3. Game Maker	34
2.5.4. Godot	35
2.6. Conclusión y justificación del proyecto	36

2.1. Cognición

Uno de los factores que separa a los seres humanos del resto de animales es su mayor desarrollo cognitivo. La cognición es la capacidad para captar, asimilar, procesar y utilizar información obtenida del entorno. Bajo este nombre se incluyen diversos procesos cognitivos como: el aprendizaje, la memoria, la toma de decisiones, la comprensión y el razonamiento.

Los primeros estudios sobre la cognición, desarrollados por Aristóteles, datan del siglo IV A.C. Aristóteles estudió el funcionamiento interno de la mente humana centrándose en la memoria y la percepción. Siempre trató de basar sus estudios en técnicas puramente empíricas mediante la experimentación y la observación. [40]

No fue hasta el siglo XV durante la Ilustración, que se retomaron los estudios sobre la mente y su funcionamiento. Y más significativamente, durante los siglos XIX y XX.

Durante la segunda mitad del siglo XIX, el auge y desarrollo de la psicología y especialmente de la psicología experimental, impulsó la investigación en el campo cognitivo. El filósofo y psicólogo alemán Wilhelm Wundt (1832-1920) desarrolló el concepto de introspección, una técnica consistente en el autoanálisis de los sentimientos de la persona, exponiéndolos de la forma más objetiva posible. [41] La introspección sentó las bases para el desarrollo de las técnicas posteriores, aunque debido a ser puramente subjetiva los psicólogos posteriores intentarán no depender de ella.

Hermann Ebbinghaus (1850-1909), otro psicólogo alemán, dedicó su carrera a la investigación de la memoria. Desarrolló un experimento con el que probar la capacidad de aprendizaje de nueva información, así como la pérdida de ella mediante el olvido. Debido a que el experimento debía evitar la intervención de cualquier conocimiento previo, Ebbinghaus creó hasta 2300 sílabas inexistentes y sin sentido para evitar ser asociadas con palabras reales. La prueba consistía en memorizar en orden aleatorio la mayor cantidad de dichas sílabas y recitarlas posteriormente. El experimento concluyó que el cerebro humano siempre intenta relacionar la nueva información con algún conocimiento anterior, y cuanto mayor es la relación encontrada mejor es la retención en la memoria de la nueva información. [42]

Los estudios de Ebbinghaus le permitieron definir el efecto de posición serial (figura 2.1): a la hora de memorizar y recitar una lista de elementos, la posición en la lista que ocupan los elementos tiene efecto en la capacidad para memorizarlos. El cerebro humano es capaz de retener mejor la información al principio

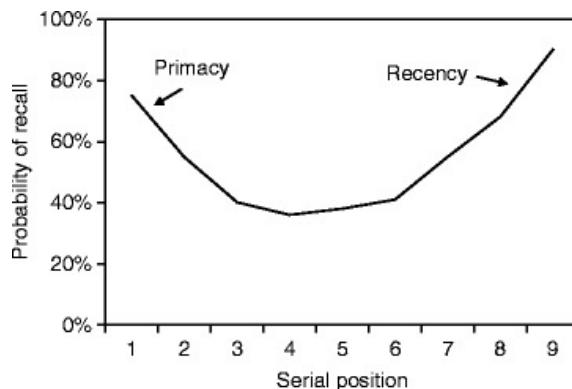


Figura 2.1: Gráfica que representa la curva creada por el efecto de posición serial. [1]

de una lista (efecto de primacía) así como al final de esta (efecto de recencia), sin embargo, los elementos centrales son que mayor esfuerzo requieren para ser memorizados.

La cognición y las habilidades y procesos cognitivos van evolucionando a lo largo de la vida de las personas.

Jean Piaget fue un psicólogo suizo que investigó el desarrollo cognitivo en el ser humano durante los primeros años de vida. Tras sus investigaciones concluyó que el desarrollo de los procesos cognitivos puede ser clasificado por etapas de crecimiento: [43]

- Etapa sensomotora (0-2 años): Existe inteligencia, pero el conocimiento se basa únicamente en las experiencias. Se desarrollan las capacidades básicas de habla y de comprensión del espacio y el tiempo.
- Etapa preoperacional (2-7 años): Se desarrolla la memoria y la imaginación. Predomina la resolución intuitiva de problemas y el pensamiento egocéntrico.
- Etapa operacional concreta (7-12 años): El pensamiento deja de ser egocéntrico y pasa a ser más sistemático y lógico. Se entienden conceptos complejos espaciales y la persona puede utilizar un proceso de pensamiento reversible.
- Etapa operacional formal (+12 años): La capacidad de pensamiento se vuelve mucho más flexible, con la posibilidad de comprender conceptos abstractos y formular hipótesis para resolver problemas complejos.

Durante la vida de una persona estas capacidades cognitivas adquiridas durante la infancia pueden resultar dañadas, degradadas o incluso perdidas com-

pletamente, dificultando o impidiendo a la persona realizar procesos como la toma de decisiones, aprendizaje o la evaluación crítica, así como llevando a la pérdida de memoria, capacidad de expresión o velocidad de procesado de información. [44]

La pérdida de estas capacidades puede estar causada o ser la causa de alguna enfermedad, por ejemplo: el deterioro cognitivo leve, en la que el paciente no puede llevar a cabo algunos procesos cognitivos de la forma esperada pero no afecta en gran medida a su vida diaria.

Trastornos más graves como la enfermedad de Alzheimer pueden llevar a peores consecuencias como la pérdida completa de la memoria inmediata, degradación en las capacidades cognitivas superiores y dificultad en la comunicación movimiento y resolución de problemas abstractos sencillos. [45]

Incluso en personas sanas, las capacidades cognitivas van disminuyendo con el tiempo, especialmente durante la vejez. Esta degradación es esperada y no conlleva grandes problemas a la vida diaria de las personas. Para frenar y limitar lo máximo posible esta degeneración es imprescindible mantener un alto nivel de actividad cognitiva. Este es el objetivo del entrenamiento cognitivo.

2.2. Entrenamiento cognitivo

El entrenamiento cognitivo, también llamado brain training, engloba al conjunto de actividades diseñadas para mantener o mejorar las capacidades cognitivas propias de una persona.

Estas actividades se basan en la neuroplasticidad humana, que significa que el cerebro es capaz de desarrollarse y transformarse en base a las experiencias vividas. Esto implica que el cerebro puede ser entrenado para fomentar el desarrollo de ciertas áreas de este involucradas con los procesos cognitivos. [46]

Las áreas objetivo de entrenamientos más usuales son aquellas con mayor relación con actividades diarias y por tanto pueden conllevar una mejora notable de las habilidades. La capacidad de memorizar, mantener la atención y la resolución de problemas son los principales procesos que entrenar.

El entrenamiento cognitivo se centra en grupos de personas con carencias cognitivas como aquellas con enfermedades neurodegenerativas como la demencia o el Alzheimer. Aunque todas las personas pueden beneficiarse del entrenamiento, especialmente personas de edad avanzada que con el paso de los años han visto mermadas sus capacidades habituales en memorización, velocidad de procesamiento, razonamiento o psicomotricidad.

El resultado del brain training puede ser muy diverso entre las personas debido a que cada una tiene un rendimiento máximo en cada habilidad. Una vez que el individuo alcanza su estado óptimo en dicha habilidad, le supondrá un esfuerzo mucho mayor continuar mejorando.

Además, hay que tener en cuenta que incluso cuando el entrenamiento mejora las capacidades cognitivas básicas, estas mejoras pueden no trasladarse a acciones más complejas, a pesar de que dichas acciones sean combinaciones de habilidades sencillas.

A día de hoy, la verdadera efectividad del entrenamiento cognitivo está en duda. Los estudios más recientes demuestran que los grandes efectos de antienvejecimiento y prevención de la enfermedad de Alzheimer son falsos y exagerados por las técnicas de marketing. Si bien el entrenamiento tiene un efecto positivo en las habilidades cognitivas, existen otros factores con mayor importancia, como la genética de cada persona. [47]

2.3. Ejercicios de Brain Training

Los ejercicios de entrenamiento cognitivo se pueden presentar en multitud de formatos, comenzando por el formato físico en papel. Los cuadernos de ejercicios han sido el medio clásico de realizar entrenamientos cognitivos. Permiten la estimulación mediante el uso de una herramienta conocida, como es el lápiz y papel, a la vez que presenta ejercicios muy variados: memoria, razonamiento, resolución de problemas o comprensión lectora, adaptados a cualquier nivel, desde niños pequeños hasta personas mayores o enfermos. Su sencillez permite que sean fáciles de diseñar, fabricar y distribuir.

2.3.1. Cuadernos de ejercicios

Los cuadernos de estimulación cognitiva de Esteve (figura 2.2) y Rubio (figura 2.3) son las más populares en España, están enfocados a personas con algún tipo de deterioro cognitivo (leve, moderado o grave) y su estructura de niveles permite una fácil clasificación y progreso en los ejercicios. [48]

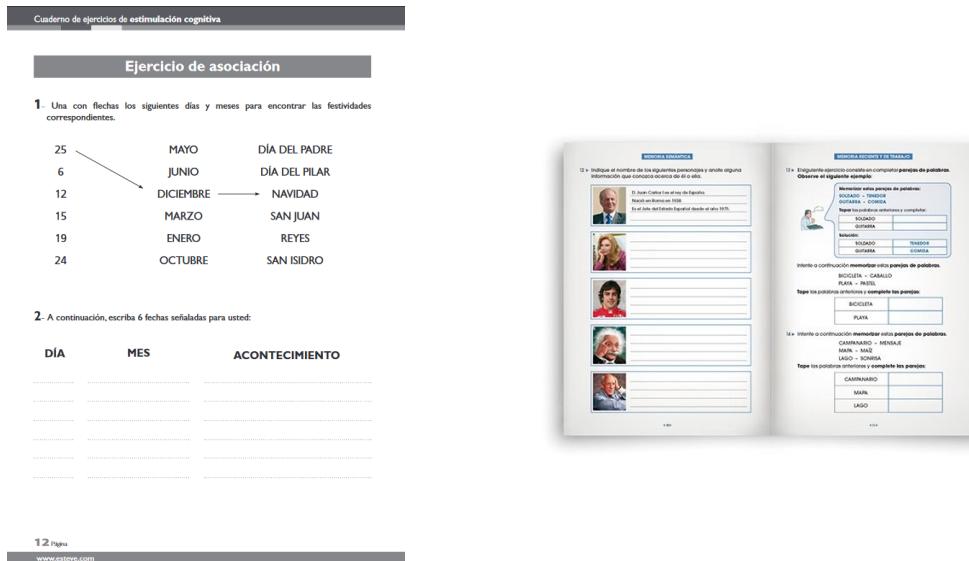


Figura 2.3: Cuaderno Rubio. [3]

Figura 2.2: Cuaderno Esteve. [2]

2.3.2. Juegos grupales

En ocasiones donde varias personas que vayan a realizar un entrenamiento cognitivo se encuentren juntas, como residencias o centros de día para mayores, se pueden realizar juegos grupales. Estos juegos permiten no solo la mejora de las capacidades cognitivas por los ejercicios si no que también impulsan las relaciones interpersonales y fomentan la creación de un ambiente amigable y alegre durante la sesión de entrenamiento. [49]

Estos juegos están diseñados para ser llevados a cabo en grupos: ya sea de forma simétrica, donde cada participante tiene un rol idéntico, o no, en los que una o varias personas ocupan un papel principal en la acción y el resto los complementa con diferentes acciones.

Existen multitud de juegos de este tipo, algunos de ellos son:

- Juego del abanico: Una persona toma el papel principal y debe transmitir al resto una emoción que elija mediante el uso únicamente de movimientos del abanico.
- Juego de expresiones: Una persona intenta expresar un sentimiento con expresiones faciales y el resto tendrá que adivinarlo.
- Juego de los nombres: Todos los participantes forman un círculo y por orden deberán decir su nombre y pasar una pelota al siguiente participante. Tras



Figura 2.4: Juego de grupo con pelota. [4]

una vuelta completa, cada persona tendrá que decir algo que le guste y lanzar la pelota a cualquier participante diciendo su nombre. Véase figura 2.4

2.3.3. Juegos y plataformas online

Con la proliferación de las nuevas tecnologías en la vida diaria de las personas, los ejercicios de estimulación se han adaptado para sacar todo el provecho que estas tecnologías ofrecen. Los ejercicios clásicos en papel que ofrecen los cuadernos son fácilmente adaptables a un entorno digital mediante el uso de aplicaciones, juegos y páginas web, que son presentados al usuario a través de un ordenador o de un dispositivo móvil como una Tablet.

Este formato es capaz de enriquecer los ejercicios ya que añade la posibilidad de personalizar cada ejercicio para el nivel y las características del usuario, así como usar animaciones, sonidos y nuevos métodos de interacción: pantalla táctil, movimiento del propio dispositivo, reconocimiento de voz, etc.

Estas plataformas online permiten un monitoreo constante de un profesional sin necesidad de estar físicamente presente con el paciente lo que puede aumentar su utilidad y eficacia frente a los cuadernos tradicionales.

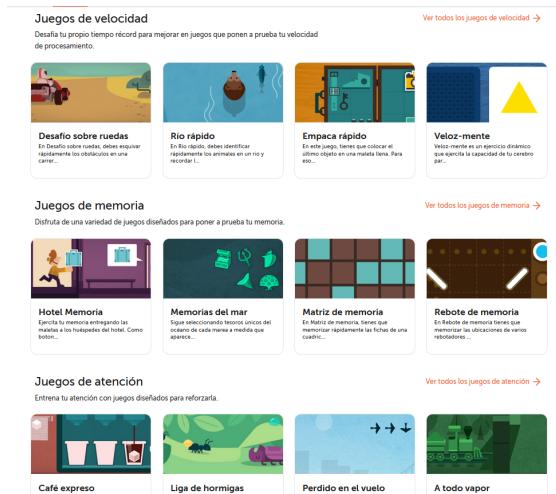


Figura 2.5: Algunos de los juegos disponibles en Lumosity.

Lumosity

Lumosity es una aplicación que reúne una serie de juegos muy llamativos para entrenar la mente de manera amena y dinámica. Se categorizan según la habilidad cognitiva que tratan: velocidad de procesamiento, memoria, atención, flexibilidad mental y resolución de problemas. En la figura 2.5 se pueden ver varios de estas categorías y sus juegos. [50]

NeuronUp

NeuronUp es una plataforma multidispositivo (figura 2.6) enfocada al uso de profesionales para monitorizar los entrenamientos de sus pacientes. Los juegos son más simples que los de Lumosity, pero ofrece una conexión directa y constante con los profesionales, que es importante para llevar un correcto entrenamiento. [51]

Brain training RV

En los últimos años se ha producido un gran auge de la realidad virtual, llegando a estar al alcance de cualquier persona con un teléfono inteligente. Los juegos de entrenamiento cognitivo han aprovechado este entorno virtual para mejorar la inmersión y realismo de sus ejercicios. Actualmente la RV está evolucionando día a día, y los juegos brain training asociados a ella también.

La RV puede usarse para mejorar ejercicios previos como los de orientación

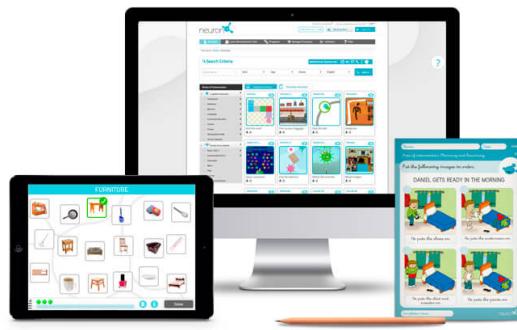


Figura 2.6: Demostración de NeuronUp multidispositivo. [5]



Figura 2.7: Juego de cálculo en Koji's Quest. [6]

espacial o los de habilidades motoras. También permite la aparición de nuevos tipos de ejercicios, especialmente aquellos relacionados con la cognición auditiva ya que en un entorno de RV se pueden generar sonidos que provengan desde cualquier punto del espacio. Además, también mejora el resto de los ejercicios y juegos ya que permite al usuario una inmersión casi completa en el entorno virtual, maximizando el impacto de cada juego.

NeuroReality

NeuroReality es una empresa que diseña software y hardware enfocado a la rehabilitación cognitiva en realidad virtual. Su principal exponente es el juego Koji's Quest (figura 2.7). Este juego está compuesto de cinco minijuegos que se adaptan al nivel de cada jugador para mantener un reto adecuado. Koji's Quest está disponible en casi todas las plataformas de RV: smartphone, Oculus Rift-/Go/Quest, HTC Vive y Google Cardboard. [52]

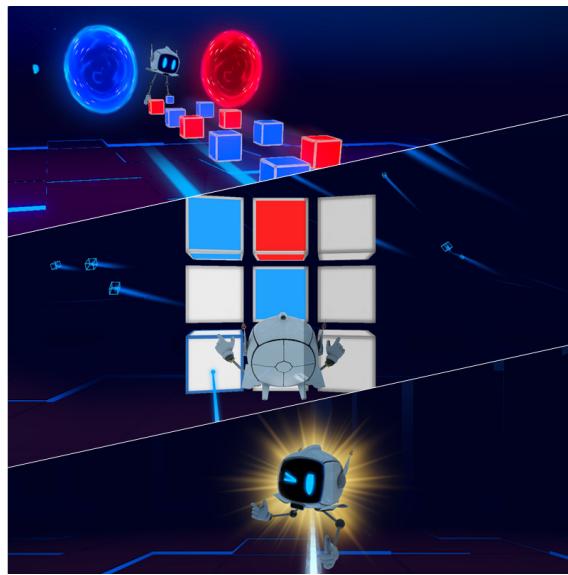


Figura 2.8: Demostración de Enhance. [7]

Virtuleap

Virtuleap ofrece su aplicación Enhance que es una compilación de juegos brain training en RV (figura 2.8). Estos juegos abarcan una amplia selección de habilidades cognitivas: memoria, resolución de problemas, orientación espacial, control motor y cognición auditiva entre otras. Su catálogo de juegos se incrementa mes a mes, lo que hace a Enhance una herramienta muy valiosa en el entrenamiento cognitivo con realidad virtual. [53]

2.4. Realidad Virtual

La realidad virtual es una tecnología que busca simular objetos, entornos o cualquier sensación de modo que parezcan reales para un usuario. Para alcanzar este objetivo se necesita que el usuario tenga la mayor inmersión posible dentro del entorno virtual, para ello se suelen utilizar gafas o cascos de realidad virtual, conocidos en inglés por las siglas HMD (Head Mounted Display).

Estos dispositivos proveen al usuario de visión estereoscópica que permite ver las imágenes recibidas en 3D, así como audio tridimensional basado en el entorno virtual u otras sensaciones hapticas mediante motores de vibración incorporados en el dispositivo.

A pesar de que los HMD son el elemento principal de la RV, tienen gran impor-

tancia también otros dispositivos que permiten la interacción del usuario con el entorno virtual, como son los mandos de juego que dan presencia en el entorno a las manos del usuario o guantes con retroalimentación que permiten a jugador sentir mediante el tacto, los objetos simulados dentro del entorno.

Actualmente la realidad virtual es una tecnología muy utilizada en una gran variedad de campos debido a la capacidad de reproducir entornos o situaciones similares a la realidad que se pueden utilizar tanto para estudio como diversión. Se utiliza por ejemplo, para simular casos médicos donde personal sanitario puede practicar cirugías sin riesgo para un paciente real, pudiendo incluso simular una representación real de un paciente concreto que previamente haya sido estudiado y escaneado para plasmar la realidad en el entorno virtual. [54]

Aunque la realidad virtual es una tecnología en alza en la actualidad, esta idea data de muchos años antes de la invención de los computadores, principalmente durante el siglo XIX en Europa, aunque el primer gran salto en la inmersión y el realismo de las obras artísticas surgió durante el Renacimiento.

2.4.1. Renacimiento

El arte de la Edad Media estaba caracterizado por su representación rudimentaria de formas tridimensionales, apareciendo completamente planas (véase la ilustración del poema 'Roman de la Rose' del siglo XIII presentada en la figura 2.9, dificultando la inmersión del espectador por su gran diferencia con el mundo en el que vive).

Esto cambió durante el Renacimiento en Europa con la introducción de la perspectiva en la pintura. La perspectiva es una técnica geométrica para representar volúmenes en un espacio 3D mediante el uso de una línea de horizonte y uno o varios puntos de fuga.

La línea de horizonte representa objetos a una distancia infinita del espectador, y el resto de los objetos de la escena escalaran en tamaño dependiendo de su cercanía al punto de vista. Las líneas paralelas a la visión del espectador convergen en alguno de los puntos de fuga existentes, creando una deformación aparente en el objeto, pero que consigue una visión más realista del mismo. Esta técnica es especialmente visible en pinturas como 'La última cena' de Leonardo Da Vinci (figura 2.10) o la 'Entrega de las llaves a San Pedro' de Pietro Perugino (figura 2.11).



Figura 2.9: Roman de la Rose. [8]



Figura 2.10: La última Cena. Leonardo Da Vinci. [9]



Figura 2.11: Entrega de las llaves a San Pedro. Pietro Perugino. [10]



Figura 2.12: Panorama de Londres. [11]

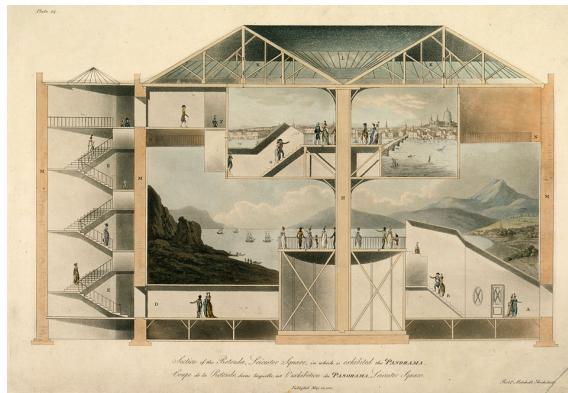


Figura 2.13: Sección de la Rotunda en Leicester Square. [12]

2.4.2. Siglo XIX

Durante el siglo XIX se populariza en Europa el uso de imágenes panorámicas que ocupan los 360º de visión horizontal del espectador para transportarlo al lugar o evento representado. [55] Comenzaron a crearse edificios de ocio dedicados a la presentación de imágenes panorámicas en los que los espectadores eran situados en el centro de una sala circular desde donde podían contemplar una o varias pinturas, normalmente de paisajes naturales, ciudades y zonas urbanas, así como grandes batallas militares, siendo una de las más populares el panorama de Londres de Robert Barker presentado en la figura 2.12.

El pintor Robert Barker fue el gran impulsor de este tipo de entretenimiento con su Rotunda en Leicester Square (véase figura 2.13), donde exponía sus pinturas.

En 1838, Charles Wheatstone, describe por primera vez como el cerebro humano procesa por separado las imágenes de ambos ojos para crear una única imagen con profundidad debido a las ligeras diferencias entre la visión de cada ojo. [56] Esto lleva al nacimiento de los primeros visores de imágenes estereoscópicas (figura 2.14). En estos visores era posible insertar una serie de diferentes tarjetas que contenían dos imágenes tomadas con una cámara estereoscópica y permitían disfrutar de su visualización con una inmersión mayor a lo común para la época.

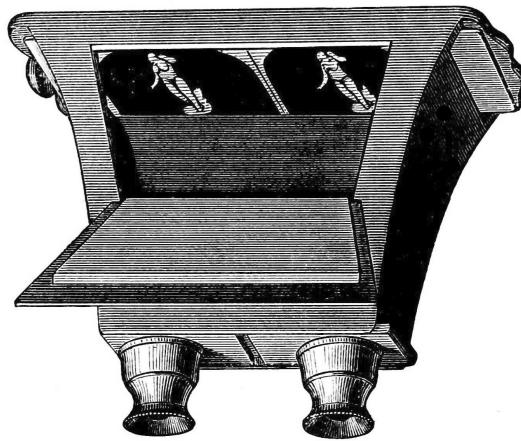


Figura 2.14: Estereoscopio de David Brewster. [13]

2.4.3. Sensorama y Headsight

En 1960 Morton Heilig creó el Sensorama (figura 2.15), el primer visor en el que se colocaba en la cabeza y permitía ver una película estereoscópica. Este dispositivo aunque limitado, estimulaba otros sentidos a parte de la visión. El sensorama disponía de una silla móvil, ventiladores para simular viento e incluso emisores de olor. Un año después, en 1961, Philco Corporation creó el primer casco de realidad virtual con sensores de movimiento, el Headsight (figura 2.16). Headsight fue desarrollado para un uso militar y permitía al usuario controlar la rotación de una cámara de vigilancia según los movimientos de su cabeza mediante el uso de sensores magnéticos que controlaban la posición del visor.

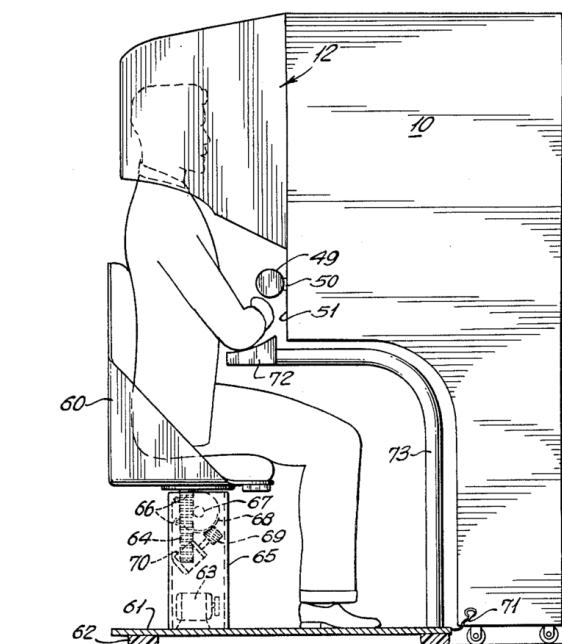


Figura 2.15: Figura 5 de la patente del Sensorama. [14]



Figura 2.16: Headsight de Philco Corporation. [15]

2.4.4. Espada de Damocles

En 1968 se dio un paso más hacia la realidad virtual que se conoce hoy en día. Ivan Sutherland y Bob Sproull tomaron la idea del Headsight y la transformaron de forma que en lugar de estar conectado una cámara de vigilancia se conectaría a un ordenador que generaría gráficos tridimensionales sencillos. Este dispositivo era tan grande y pesado que necesitaba ser sostenido del techo sobre el usuario, dándole así su nombre en referencia a la leyenda de Damocles.

en la cultura griega clásica. En la figura 2.17 se puede comprobar el tamaño del dispositivo y su colocación sobre el usuario. [57]

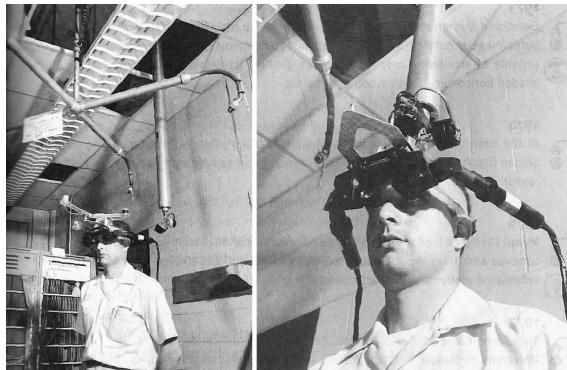


Figura 2.17: Espada de Damocles de Ivan Sutherland y Bob Sproull. [16]

2.4.5. Realidad virtual

A pesar de todos estos avances en la reproducción de una realidad artificial, no se disponía de un nombre y fue en 1987 cuando el término realidad virtual es definido por Jaron Lanier un informático y compositor estadounidense. Tras fundar una compañía llamada Visual Programming Lab, diseñó y desarrolló los que serían los primeros dispositivos de realidad virtual disponibles para ser adquiridos por el público: el EyePhone y el Dataglove, un visor y un guante para interactuar con el espacio virtual. [58]

Pese al gran avance que suponen estos productos, la tecnología aún tenía mucho por mejorar. El sistema del EyePhone junto con el ordenador necesario para generar las imágenes del mundo virtual costaban hasta 250.000 dólares, y aun así solo eran capaces de alcanzar una tasa de imágenes de 5 fotogramas por segundo, una cifra insuficiente para ofrecer una experiencia fluida al usuario. [59]

Este visor contenía elementos nunca vistos en el ámbito de la realidad virtual. Utilizaba sensores capaces de reconocer las expresiones faciales del usuario, todos los sonidos eran procesados por el ordenador para darles un efecto 3D de forma que su fuente se movía por el espacio dependiendo de los movimientos de la cabeza del usuario. El Dataglove solo era capaz de transmitir sensaciones hapticas al usuario si no que también actuaba como sensor, pudiendo generar en el espacio virtual un modelo de la mano del usuario que éste podía utilizar para interactuar con el ambiente. [60]

A pesar de que este dispositivo fue el primero que se asemejaba a la visión

actual de un visor de realidad virtual, requería de una gran potencia informática para la época, resultando en un producto demasiado caro y voluminoso, como se puede ver en la figura 2.18.



Figura 2.18: Eyephone y Dataglove junto al sistema de seguimiento Polhemus en exposición Nissho Iwai en Tokio en 1999. [17]

2.4.6. Virtual Boy

En 1995 la compañía de videojuegos japonesa Nintendo lanza al mercado su Virtual Boy, la primera consola portátil con gráficos verdaderamente tridimensionales (figura 2.19). Esta consola prometía ser un gran éxito, pero acabó siendo un fracaso. Su precio, aun siendo más asequible que el EyePhone, seguía siendo elevado para lo que ofrecía: 180 dólares. [61]

La Virtual Boy disponía de dos matrices lineales de láser de 1x224 que escaneaban constantemente el campo visual de cada ojo con la ayuda de dos espejos que oscilaban rápidamente. Por esto, la consola emitía un constante murmullo mientras estaba en uso.

La consola solo podía reproducir un único color, junto con el negro. Se escogió el color rojo por las características de los LED de este color: son más baratos y consumen menos energía que otros colores.

Esta consola utilizaba un procesador NEC V810 de arquitectura RISC y 32 bit a 20MHz, que era acompañado de 1MB de memoria RAM. La resolución efectiva de las pantallas era de 384x224 píxeles. Toda la consola era alimentada a partir de 6 pilas AA situadas en el mando. [62]

Por desgracia, el uso prolongado de este dispositivo causaba dolores de cabeza y mareos, que, junto con su pequeña y básica selección de juegos (entre ellos, el más popular: Mario's Tennis, representado en la figura 2.20), llevaron a



Figura 2.19: Virtual Boy de Nintendo. [18]

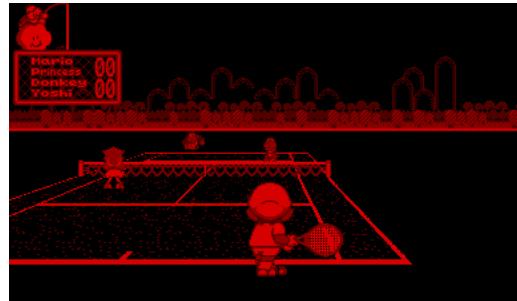


Figura 2.20: Mario's Tennis. Videojuego de Nintendo para Virtual Boy. [19]

que la consola fuera descontinuada menos de un año después de su lanzamiento.

2.4.7. Siglo XXI

En los últimos años ha habido un crecimiento acelerado de la tecnología: los ordenadores han multiplicado su potencia y capacidad de procesamiento paralelo, los dispositivos móviles se han vuelto mucho más sofisticados que cualquier ordenador del siglo XX, las pantallas son capaces de reproducir millones de colores a altas frecuencias de refresco y el acceso a Internet de alta velocidad es algo común. Estos avances han hecho resurgir con más fuerza que nunca la realidad virtual, especialmente en el ámbito de los videojuegos.

En 2010, el joven empresario Palmer Luckey creó el prototipo de un casco de realidad virtual al que llamó Oculus Rift, véase la figura 2.21. Este dispositivo podía representar imágenes de alta resolución con un campo de visión de 90º, algo muy superior a lo existente hasta el momento. [63] Dos años después, Oculus Rift logró financiación a partir de una campaña de micro financiación.

Entre los años 2014 y 2015 se produjo un boom en la realidad virtual de consumo, apareciendo muchas alternativas para el público general:

- Sony anuncia PlayStation VR un casco de RV para su consola PlayStation 4. Figura 2.22.
- Samsung lanza al mercado su Samsung Gear VR, un visor que utiliza un



Figura 2.21: Primer kit de desarrollo de Oculus Rift. [20]



Figura 2.22: PlayStation VR. [21]

smartphone (Samsung Galaxy) como centro de procesamiento y pantalla.

- Google produce las Google Cardboard, una alternativa barata que permite al usuario crear su propio visor y utilizar su propio dispositivo móvil con Android como pantalla. Figura 2.23.

Con la aparición de dispositivos capaces de reproducir contenido de realidad virtual, también fueron apareciendo videojuegos y otras experiencias para los mismos. La RV era ya una parte fundamental del ocio y los videojuegos.

En 2016 se lanza al mercado HTC Vive, un completo sistema de realidad virtual (véase figura 2.24) producido en colaboración entre HTC y Valve, productores de hardware y software respectivamente. El HTC Vive pasó a ser el casco de RV de referencia en el sector: contaba con pantallas OLED de alta resolución (1080x1200 píxeles) y tasa de refresco de 90Hz que ofrecían un campo de visión



Figura 2.23: Google Cardboard ensamblada. [22]



Figura 2.24: HTC Vive y todos sus complementos. [23]

de 110º, contaba con dos mandos de control táctiles con respuesta háptica, así como un sistema de sensores colocados en la habitación que permitían definir un espacio de juego de hasta 25 m² en el que el jugador podía moverse libremente. [64]

2.4.8. Actualidad

Actualmente, la realidad virtual continua sin ser perfecta pero también continua mejorando cada dia. En el año 2016 Sony lanzó al mercado su apuesta por la RV con PlayStationVR. Este dispositivo tuvo una gran aceptación entre el público debido a su menor precio y especialmente a la enorme acogida de la consola PlayStation 4. Durante varios años ha sido el dispositivo de VR más vendido en el mercado con 6,6 millones de unidades vendidas entre 2016 y 2021. [65]

Más recientemente, se han originado dos enfoques para la RV claramente diferenciados y liderados por dos empresas principalmente: Valve y Meta.

En verano de 2019 Valve lanza al mercado un nuevo producto. Tras romper su



Figura 2.25: Kit de visor Valve Index junto a controles y estaciones base. [24]

alianza comercial con HTC, Valve desarrolla su propio visor de RV llamado Valve Index, mostrado en la figura 2.25. Este dispositivo mejora en todos los aspectos a su predecesor, el HTC Vive: [66]

- Dispone de pantallas OLED de 1440x1600 píxeles con refresco de hasta 144Hz.
- Dos cámaras estereoscópicas situadas al frente del dispositivo permiten su uso durante el juego como cámaras de video o como sensores.
- Los mandos de control han sido mejorados y ahora son capaces de detectar movimientos de cada dedo individualmente gracias a sensores de proximidad.
- Las nuevas estaciones (sensores colocados en la habitación) ahora permiten un campo de juego de hasta 100m² comparados con los 25m² del HTC Vive.

Pero todas estas mejoras conllevan un gran coste y el precio del kit completo asciende hasta 1080 euros en su lanzamiento. A este precio habrá que sumar el precio de un ordenador con capacidad suficiente para ejecutar los videojuegos tan demandantes que están diseñados para este dispositivo.

Por otra parte, Meta lanza en 2020 su nuevo producto, pero con una mentalidad completamente diferente: en lugar de buscar la máxima potencia e innovación tecnológica, se busca una mayor portabilidad y eficiencia manteniendo el coste bajo para que sea accesible por un mercado más amplio. Este nuevo dispositivo es el Meta Quest 2, sucesor del anterior Oculus Quest. Este visor no se conecta a un ordenador para reproducir los videojuegos, si no que se trata de un visor de RV completamente independiente. Véase figura 2.26.

El Meta Quest 2 contiene un sistema Android personalizado que se ejecuta sobre un SoC (System on a Chip) Snapdragon XR2, que está acompañado de



Figura 2.26: Meta Quest 2. [25]

6GB de memoria RAM y hasta 256GB de almacenamiento. Tiene una única pantalla LCD que ofrece una resolución de 1832x1920 píxeles por cada ojo, y que funciona hasta a 120Hz en ciertas casos, aunque normalmente trabaja a 90Hz. [67]

Este dispositivo no depende de estaciones de sensores fijas en la habitación ya que es capaz de generar la zona de juego a partir de los sensores que lleva incorporados el propio visor. Debido a esto aparecen nuevas restricciones en los mandos: para que estos funcionen de forma adecuada deberán estar situados delante del visor, si el usuario hace un movimiento demasiado amplio, los mandos saldrán de la zona captada por los sensores y dejarán de ser reconocidos.

Pese a ser un visor más limitado que el Valve Index, la portabilidad y precio del Meta Quest 2 hacen que sea uno de los visores de RV más para tener en cuenta en la actualidad, y el más vendido con 10,4 millones de unidades, en comparación con las 600.000 unidades vendidas de Valve Index. [65]

2.5. Motores de videojuegos

Para diseñar y desarrollar videojuegos, ya sean de tipo brain training o cualquier otro, se utiliza habitualmente un motor de juego.

Un motor de juego es un entorno de trabajo o framework que contiene funcionalidades básicas para la creación de videojuegos como: un motor de físicas, renderizado de gráficos, gestor de comunicación en red o comunicaciones a bajo nivel con el hardware en el que se ejecutará el videojuego. Por tanto, un motor de juego provee de una base sobre la que desarrollar un videojuego de forma más ágil, ahorrando tiempo, esfuerzo y dinero. [68]

2.5.1. Unity

Es un motor capaz de crear videojuegos multiplataforma de multitud de géneros de manera sencilla. Es el motor más utilizado por desarrolladores independientes debido a su bajo coste y a su gran comunidad de desarrolladores online y su gran documentación. [69] Figura 2.27.

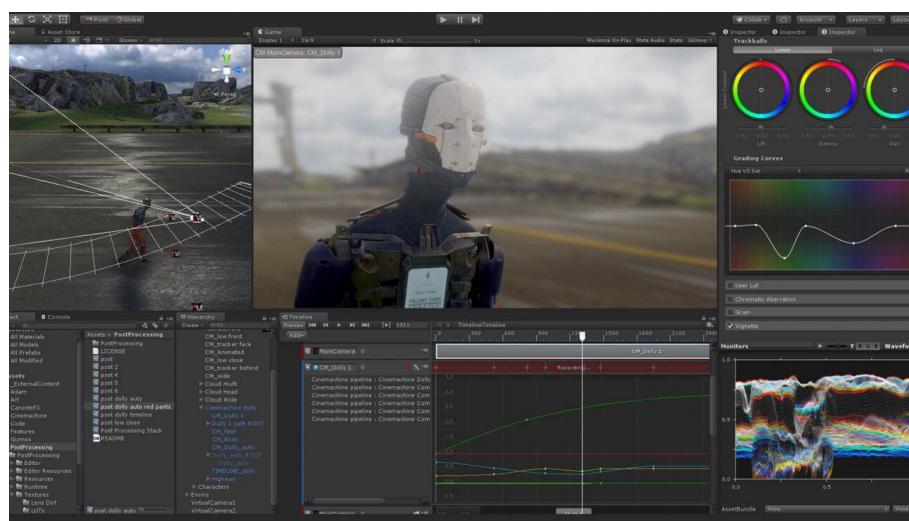


Figura 2.27: Interfaz de Unity. [26]

Destaca en la creación de videojuegos para plataformas móviles. Con este motor se ha desarrollado juegos como *Ori and the Blind Forest*, *Hollow Knight*, *Monument Valley 2* (figura 2.28), *Kerbal Space Program* o *Firewatch*.

Unity cuenta con un plan personal completamente gratuito y libre de regalías. Pero en caso de que la recaudación del juego supere los 100.000 dólares al año

se deberá adquirir un plan de pago anual de entre 369 y 4554 dólares. En caso de superar los 200.000\$ de ingresos o contar con más de 3 trabajadores, se deberá contratar un plan empresarial propuesto por Unity. [70]

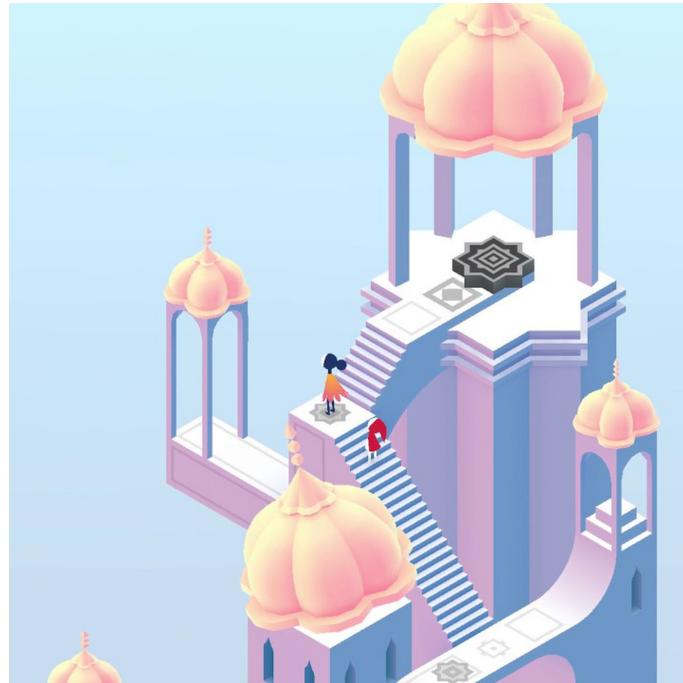


Figura 2.28: Monument Valley 2. [27]

2.5.2. Unreal Engine

Actualmente es uno de los motores más utilizados por su potencia y acabados gráficos, así como permitir trabajar para distintas plataformas, entre ellas PC, PS4, Xbox One, Nintendo Switch, Android o iOS. Fue creado por la empresa Epic Games, desarrolladora de juegos como Fortnite y la serie Unreal. Figura 2.29.

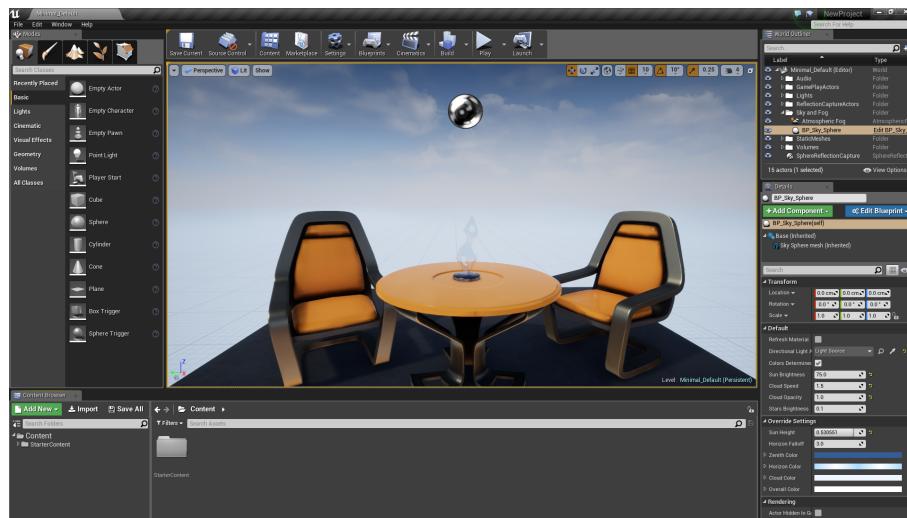


Figura 2.29: Interfaz de Unreal Engine 4. [28]

Unreal Engine permite un elevado grado de personalización, lo que hace que pueda usarse para crear juegos de multitud de géneros diferentes: Aventura, FPS, lucha 2D, puzzles, etc. Este motor ha sido utilizado para crear los videojuegos de la saga Gears of War, Bioshock, Kingdom Hearts III y Fortnite (figura 2.30) entre muchos otros. [71]

Este motor presenta un modelo de negocio basado en regalías, de modo que su uso es completamente gratuito, pero a cambio Epic Games recibirá un 5 % de los ingresos del juego en caso de que este supere el millón de dólares en ventas totales. [72]



Figura 2.30: Visión del jugador durante una partida de Fortnite. [29]

2.5.3. Game Maker

Este motor tiene la particularidad de que no requiere conocimientos extensos en programación, si no que permite usar una interfaz para seleccionar y arrastrar una serie de acciones predefinidas y de esta forma crear los eventos que formarán el juego. Además de esta interfaz, también permite el uso de su propio lenguaje GML para la programación en profundidad de los eventos. Estas características lo hacen un motor ideal para aquellas personas con conocimientos limitados en el campo de la informática. [73] Figura 2.31.

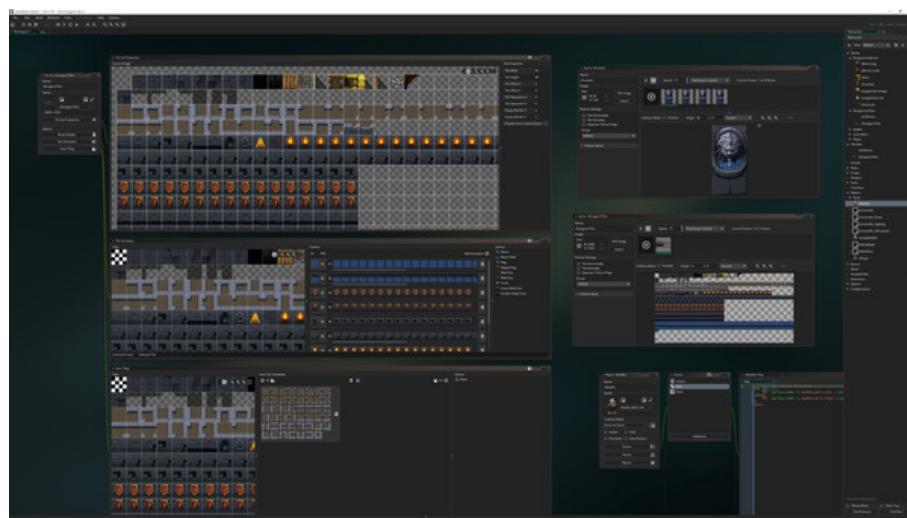


Figura 2.31: Interfaz de GameMaker Studio 2. [30]

Sin embargo, este motor es mucho más limitado que los anteriores en la lista, tanto en capacidad y rendimiento, como en variedad de géneros que puede abarcar, estando limitado a los gráficos 2D o 3D muy básicos. A pesar de estas limitaciones, su versión más reciente permite la distribución en prácticamente todas las plataformas habituales en la actualidad.

GameMaker sigue un modelo de pago anual con diferentes planes que se diferencian principalmente en las plataformas disponibles como destino del juego. Se puede utilizar de forma totalmente gratuita aunque solo permite exportar juegos para el navegador OperaGX. Los planes de pago van desde los 50\$ al año para la versión más básica que solo permite publicar en una plataforma: Windows o MacOs, hasta los 800\$ al año que permite publicar juegos para cualquier plataforma, incluyendo consolas como PlayStation 5, Xbox Series X/S y Nintendo Switch. [74]

Debido a su facilidad de uso y la no necesidad de conocimientos profundos



Figura 2.32: Captura de pantalla de Crashlands. [31]

de informática, es un motor que muchos desarrolladores independientes utilizan para sus juegos. Por ejemplo, para juegos como Crashlands (figura 2.32), Hyper Light Drifter o Forager.

2.5.4. Godot

Este motor es el único de la lista que es completamente gratuito y de código abierto. Permite crear todo tipo de videojuegos tanto en 2D como en 3D y además tiene una gran comunidad de desarrolladores en Internet, haciéndolo uno de los motores más fáciles de usar a la vez que versátiles. [75] Figura 2.33.

Godot engine soporta la creación de juegos para PC y dispositivos móviles como Oddventure o RivenTails: Defense (figura 2.34), sin embargo, su soporte para consolas de sobremesa es limitado y no oficial.

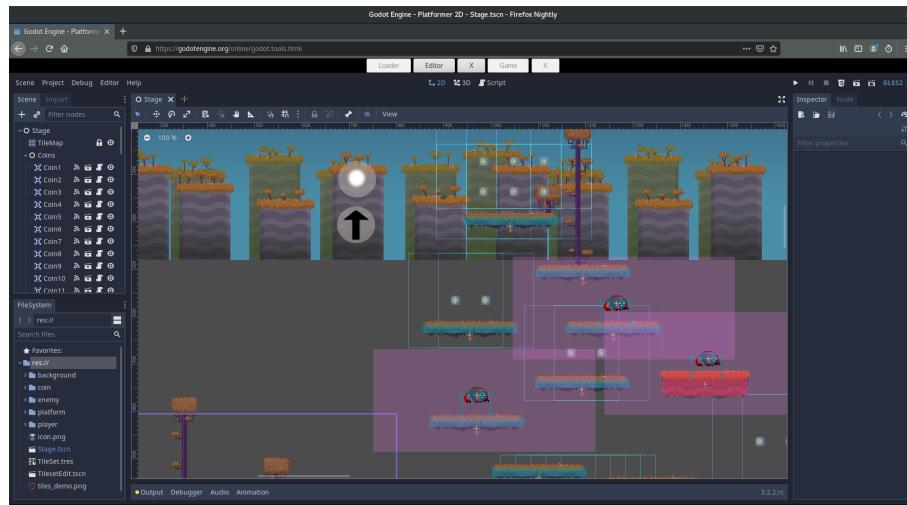


Figura 2.33: Interfaz de Godot en modo web. [32]



Figura 2.34: Captura de pantalla de RivenTails: Defense. [33]

2.6. Conclusión y justificación del proyecto

La importancia de los ejercicios de brain training para mantener en buen estado las capacidades cognitivas de las personas es evidente, y por ello estos ejercicios han ido evolucionando y cambiando para adaptarse a los distintos medios disponibles. En la actualidad se puede acceder a estos ejercicios desde multitud de dispositivos desde la comodidad del salón. Sin duda esto ha supuesto

una mejora en la calidad de vida de muchas personas y de una forma amigable y entretenida, ya que las nuevas tecnologías no solo facilitan el acceso a los ejercicios, si no que los pueden mejorar en gran medida.

El uso de vídeos, programas interactivos o animaciones coloridas resulta en ejercicios mucho más atractivos para el usuario, que no los verá como una carga o un proceso tedioso, si no como un juego entretenido y divertido. A pesar de esto, los ejercicios actuales tienen raíz en los ejercicios tradicionales y son una evolución directa de los mismos, heredando algunas de sus limitaciones.

La evolución de la realidad virtual permite que hoy en día puedan crearse experiencias mucho más inmersivas e interactivas que cualquier otra tecnología que utilice una pantalla y métodos de control normales. Aunque en la actualidad existen algunas pruebas utilizando RV aplicada al entrenamiento cognitivo, sigue siendo un campo sin aprovechar completamente.

Es especialmente importante destacar que el entrenamiento cognitivo tiene un componente lúdico que es muy importante aprovechar para mantener al usuario motivado y con ánimo. De igual forma, la realidad virtual no es algo puramente recreativo; puede usarse y se usa con otros propósitos, especialmente en medicina, donde se pueden utilizar modelos 3D de pacientes reales para diagnosticar enfermedades o evaluar intervenciones quirúrgicas incluso antes de realizarlas.

Por este motivo, este proyecto busca complementar la parte lúdica de los ejercicios con el componente serio de la realidad virtual, intentando crear un videojuego que sea capaz de sacar todo el partido a ambos campos.

Por supuesto, esto no sería posible sin la proliferación y popularidad del videojuego, que lo ha convertido en un medio de gran importancia, dando lugar a la creación de motores de videojuegos disponibles para el público, que permiten el desarrollo de videojuegos independientes.

Capítulo 3

Análisis inicial del problema

Contenido

3.1. Pruebas	40
3.1.1. Motricidad	40
3.1.2. Memoria	42
3.1.3. Lenguaje	43
3.1.4. Razonamiento	44
3.1.5. Comprensión espacial	45
3.2. Desarrollo	46

Para alcanzar el objetivo de este proyecto se va a desarrollar un videojuego del género de concursos, en el que el jugador tendrá que resolver una serie de pruebas de diferentes categorías. En el desarrollo de videojuegos se suele crear un Documento de Diseño de Juego (GDD por sus siglas en inglés) de forma previa al inicio del desarrollo. En éste se documentan todas las ideas y planificaciones para tener en cuenta durante el desarrollo del videojuego, incluyendo personajes, historia, mecánicas, público objetivo o plataforma.

Para el desarrollo de este proyecto también se ha creado un GDD, que se encuentra añadido a esta memoria en el apéndice **Game Design Document (A)**.

Este videojuego es ante todo un juego serio, es decir, su principal propósito no es el de entretener, si no el de permitir al jugador entrenar sus habilidades cognitivas. Eso no quiere decir que el juego no contenga una parte lúdica destinada a entretenir al usuario y hacer el juego divertido. En este caso, el videojuego estará ambientado en un plató de televisión para concursos similar al mostrado



Figura 3.1: Plató del programa de televisión 'Atrapa un millón'. [34]

en la figura 3.1, tomando una estructura similar a dichos concursos.

El jugador se encontrará en el centro de un plató, rodeado de público y con una pantalla en frente en la que se le presentarán pruebas y preguntas. Dependiendo del resultado de cada prueba, el jugador sumará puntos a su marcador, que servirá para valorar su sesión de entrenamiento al final del juego. A pesar de utilizar un sistema de puntuación, no se trata de un juego puramente competitivo, la función de los puntos es tratar de animar y recompensar el jugador.

3.1. Pruebas

Las pruebas están diseñadas para estimular determinadas habilidades cognitivas ya vistas en el apartado 2.1 a partir de los juegos y mecánicas que se utilizan en entrenamientos cognitivos actuales como los presentados en la sección 2.3. Estas pruebas se pueden agrupar en cinco categorías dependiendo principalmente del tipo de habilidades cognitivas que ejercitan.

3.1.1. Motricidad

En esta categoría se incluyen las pruebas que se centran en entrenar la capacidad de movimiento, reflejos motores y la conciencia de la posición del cuerpo de uno mismo en el espacio. Estas pruebas requieren del movimiento del jugador por lo que dependiendo de las capacidades individuales de los usuarios pueden resultar más o menos complejas.

Se busca, además, activar físicamente al jugador, no solo para mantener una buena salud cognitiva, si no también física.



Figura 3.2: Boceto del escenario para la prueba de baile.

Baile

La música es uno de los estímulos más potentes para el ser humano, por lo que en este caso se pretende usarla para animar al jugador y hacer que se mueva al ritmo de canciones conocidas por el mismo. El jugador tendrá que realizar una serie de movimientos con sus brazos y piernas al ritmo de la música.

Estos movimientos requerirán posicionar una o varias extremidades en lugares concretos relativos al usuario, realizar movimientos de un punto a otro como se ve en la figura 3.2, o se dejará libertad para que el usuario elija sus propios movimientos durante un tiempo. Con esto último se pretende que el usuario se deje llevar por los sentimientos evocados por las canciones, sirviendo como ejercicio de ocio y expresión emocional.

Figuras

De forma similar a la prueba anterior, el jugador debe adquirir una serie de poses concretas, aunque en este caso se realizará sin música. Eliminando la música se obtiene una prueba que puede ser menos atractiva para el usuario, pero que permite añadir complejidad a las poses y movimientos a ejecutar, puesto que no hay un ritmo que seguir y el usuario está en un estado más calmado. De esta

forma el jugador puede tomarse el tiempo que necesite antes y después de cada figura.

Parada

Esta prueba se basa en las pruebas anteriores, pero añadiendo un modificador para entrenar los reflejos del jugador. Durante la prueba el jugador deberá realizar una serie de movimientos amplios y continuos, buscando realizar movimientos fuera del rango usual para evitar la pérdida de flexibilidad en las articulaciones. Cuando reciba una señal, deberá detener el movimiento de inmediato y mantener la posición hasta que se dé la siguiente señal para continuar el movimiento.

Objetivos

La prueba de objetivos se enfoca en los reflejos y la capacidad del jugador de conocer y entender la posición de sus extremidades en el espacio. En la prueba una serie de objetivos irán apareciendo en el campo visual del jugador. Estos objetivos se moverán desde el frente del jugador hacia él y este deberá tocarlos con sus manos para hacerlos desaparecer.

3.1.2. Memoria

Esta categoría trata de entrenar la memoria a largo plazo del jugador, para ello utiliza los dos sentidos más fuertes y que provocan recuerdos más arraigados en la memoria: la visión y el oído.

Canción

Esta prueba utiliza la música para hacer recordar al jugador y evocar memorias que pueda haber asociado a la canción. Para ello es importante que la canción sea conocida por el usuario, por lo que es muy importante tener en mente el público objetivo del videojuego, en este caso, personas mayores. Durante el desarrollo de la prueba se presenta al jugador la canción y este tendrá que averiguar el autor y nombre de la canción, así como posiblemente parte de su letra o relatar experiencias y recuerdos que pueda tener.

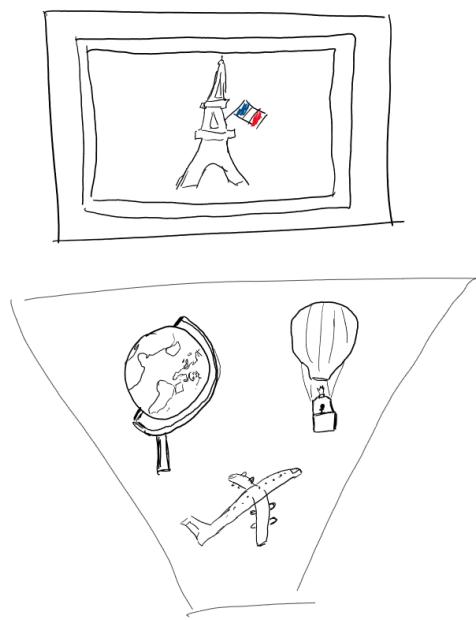


Figura 3.3: Boceto de la prueba de turismo. Se presenta una imagen de la torre Eiffel con decorado extra al frente.

Turismo

La prueba de turismo se basa en el mismo principio que la anterior, pero en lugar de utilizar una estimulación acústica, se utiliza una visual. En concreto, presentando imágenes de monumentos o ciudades famosas a nivel mundial con las que el jugador pueda estar familiarizado (figura 3.3). Se intenta que el jugador recuerde o descubra de qué monumento se trata y dónde se encuentra, añadiendo si fuera posible, anécdotas o experiencias propias relacionadas.

3.1.3. Lenguaje

Esta categoría se centra en entrenar el razonamiento y el habla del usuario. Para ello se presentarán imágenes que para que el jugador piense y hable sobre lo que sucede en ellas, posiblemente también activando la memoria del usuario en caso de que haya experimentado una situación similar.

Situaciones

En esta prueba el estímulo visual que recibe el jugador son imágenes de situaciones cotidianas con las que el jugador pueda, posiblemente, identificarse.

Se intenta hacer que el usuario hable y desarrolle un tema lo máximo posible, pudiendo incluso responder preguntas sobre la situación planteada. Por esto, las imágenes deben representar situaciones habituales o claramente reconocibles, pero intentando mantener un valor emocional que pueda despertar recuerdo en el usuario, entrenando así su memoria, dando pie a más desarrollo por su parte e incluso a revivir ciertas sensaciones y sentimientos.

Descripciones

Esta prueba es análoga a la anterior, pero intentando llevar el proceso a la inversa. En este caso se presentarán una serie de descripciones textuales al usuario, a partir de las cuales él deberá inferir qué está sucediendo en la hipotética situación. Por ejemplo, se pueden describir el aspecto de personas o cómo van vestidas, qué objetos se encuentran al rededor, o qué se puede oír en la escena. De este modo se puede entrenar no solo la capacidad de razonamiento y las habilidades cognitivas de la prueba anterior, si no también se practica la lectura y la comprensión lectora.

3.1.4. Razonamiento

Las pruebas de esta categoría se centran en entrenar el razonamiento y la percepción de objetos y sonidos. Estas pruebas utilizan de forma más prominente elementos de realidad virtual, presentando al usuario objetos tridimensionales con los que puede y debe interactuar. El jugador puede coger, mover y observar de cerca cualquier objeto de estas pruebas. Se busca así alimentar la curiosidad del jugador y potenciar su razonamiento sobre los objetos que tiene a su disposición.

Agrupación de objetos

Delante del jugador aparecen una serie de objetos modelados en 3D (véase figura 3.4). Estos objetos son completamente interactivos y el jugador tiene que intentar clasificarlos en dos categorías. Las categorías serán fijas para cada prueba, pero el jugador no las conocerá, por lo que tendrá que razonar para encontrar una conexión entre dos pares de objetos.

Es importante que los objetos elegidos representen su categoría de forma inequívoca y que habiendo presentes dos objetos de una misma categoría, su conexión sea aparente. Una vez el jugador descubra cuáles son las dos categorías ocultas y qué par de objetos pertenece a cada una, deberá coger los objetos

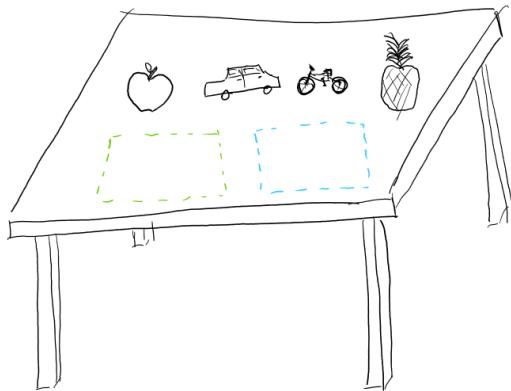


Figura 3.4: Boceto para la prueba de asociación de objetos, mostrando dos zonas (verde y azul) para la clasificación de los objetos.

virtuales y moverlos para posicionarlos en una zona determinada para cada categoría.

Asociación de sonidos

En esta prueba se presentan al jugador cuatro objetos virtuales de la misma forma que en la prueba anterior, pero en este caso no tendrá que clasificarlos. Todos los objetos representarán análogos del mundo real que produzcan sonidos fácilmente reconocibles. A continuación, se reproduce para el usuario el sonido característico de uno de los objetos que tiene delante. El jugador tendrá que razonar y recordar de sus experiencias en la vida real cuál de los objetos es el que produce el sonido que está escuchando. Cuando está seguro de su respuesta, el jugador coge el objeto virtual y lo selecciona como su respuesta.

3.1.5. Comprensión espacial

En esta última categoría se intenta usar las ventajas de la realidad virtual para entrenar la comprensión espacial del usuario. Gracias a la RV el jugador puede moverse libremente dentro del espacio de juego, coger y mover objetos virtuales, y observarlos de cerca, todo ello con una curiosidad despertada por el formato de la RV que es uno al que la mayoría de los jugadores no estarán acostumbrados. Utilizando estas características se han diseñados dos pruebas que son especialmente efectivas en RV por su grado de inmersión y las posibilidades que ofrece y que, de intentar imitarlas en la vida real, no serían tan efectivas o romperían la inmersión en el entrenamiento cognitivo.

Figuras superpuestas

En esta prueba se presentan al jugador varias siluetas oscurecidas, posiblemente de objetos tridimensionales usados en otras pruebas, para que se trate de descubrir de qué objeto se trata en cada caso. También se pueden presentar objetos no oscurecidos, pero superpuestos visualmente unos con otros. El jugador tiene que entender la posición que ocupa cada objeto en el espacio, separándolos mentalmente para descubrir qué objetos son los que tiene delante.

Localización de sonidos

Las personas tienen dos oídos, uno a cada lado de la cabeza, lo que produce ligeras diferencias en el sonido que capta cada uno. Este comportamiento es análogo al de la vista, en el cual se basa toda la realidad virtual (véase párrafo sobre la visión estereoscópica en el apartado 2.4.2). Por ello, las personas son capaces de distinguir la procedencia de un sonido incluso con los ojos cerrados. Durante mucho tiempo no se ha dado importancia a esta faceta del audio digital, pero con el auge de la RV es cada vez más común que videojuegos utilicen audio espacializado digitalmente para imitar el comportamiento del mundo real. Unity soporta este tipo de audio, por lo que se puede crear una prueba que lo aproveche y sea capaz de entrenar la comprensión espacial del jugador. Durante esta prueba, se coloca una fuente de sonido espacializado en algún punto al rededor del usuario (vease figura 3.5), y este será capaz de percibir desde qué punto proviene el sonido. Para superar la prueba el jugador debe descubrir dónde se encuentra la fuente sonora y girarse para mirarla de frente.

3.2. Desarrollo

Antes de comenzar el juego, se presentará al usuario una pantalla en la que podrá elegir ciertos parámetros para el juego y que servirán para adaptar la sesión de juego al jugador concreto. Estas opciones ajustan el tipo de pruebas que pueden aparecer o no durante la partida, el número de rondas a jugar, si el jugador estará sentado o de pie, si tiene alguna dificultad auditiva o motora, o si la realización de las pruebas otorgará puntos, entre otros parámetros. El objetivo de esta pantalla de configuración es poder crear la mejor experiencia de juego posible para cada usuario.

A continuación, el juego dará comienzo y el jugador aparecerá en un entorno virtual ambientado en un platón de televisión donde se realizará la introducción al

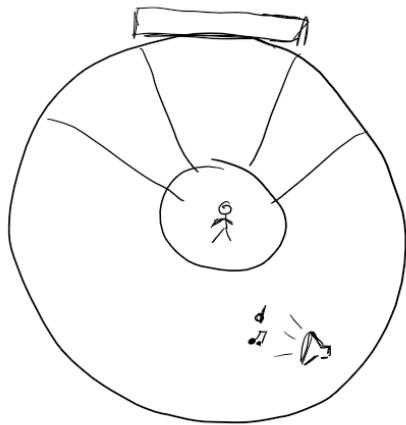


Figura 3.5: Boceto de la vista cenital del escenario principal, mostrando una posible colocación de la fuente de sonido para la prueba de localización de sonidos.

juego. Una vez todo listo dará comienzo la primera ronda del concurso. Todas las rondas siguen la misma estructura:

- Primero se presentarán al jugador dos pruebas diferentes y este podrá elegir a cuál de ellas va a enfrentarse en la ronda.
- A continuación, el jugador será transportado a un nuevo entorno propiamente ambientado para la prueba que ha elegido. Cuando esté listo, comenzará la prueba.
- Si las puntuaciones están activas, el jugador recibirá una puntuación a cor-te a su ejecución en la prueba siempre que sea posible cuantificar dicha ejecución.
- Finalmente, el jugador es transportado de vuelta a escenario de juego principal y la ronda finaliza.

Tras la finalización de la última ronda, se realiza un recuento de puntos si procede, se muestra un resumen de las actividades realizadas y se da por terminada la sesión de juego.

Este juego estará diseñado para que pueda ser utilizado por una persona de manera individual, pero también permitirá la interacción de una persona externa al entorno virtual. Esta persona actuará como ayudante del jugador, pudiendo cambiar el comportamiento del juego saltando pruebas, evaluando y puntuando las mismas o simplemente guiando al jugador, que puede verse descolocado en

el entorno virtual. Además, por la naturaleza de algunas de las pruebas, se requiere de la interacción y evaluación de una persona con los conocimientos adecuados en entrenamiento cognitivo. De esta manera se ofrece este videojuego como una herramienta para que profesionales puedan animar a las personas que reciben entrenamiento cognitivo y hacerlo más llevadero y entretenido.

Capítulo 4

Tecnología a usar

4.1. Tecnología a usar

Para el desarrollo de este proyecto se necesita un dispositivo de realidad virtual simple y compacto, que sea fácil de transportar y que no requiera de mucha preparación para utilizarlo, ya que será usado por personas mayores o con movilidad limitada y es necesario que el dispositivo no suponga ningún impedimento a la hora de utilizarlo. Por esto se han descartado los dispositivos con cables que los conectan a un ordenador o aquellos que necesitan de sensores externos y que complicarían la instalación y uso.

Además de cumplir estas características, es necesario que sea una plataforma soportada por algún motor de videojuegos de los que están preparados para desarrollar aplicaciones de realidad virtual.

Los dos principales cascos de RV que cumplen estas características son el Oculus Go y el Meta Quest 2 (véase figura 4.1). Ambos dispositivos son inalámbricos y funcionan con batería. Cuentan con una distribución especializada de Android sobre la que se ejecutan los juegos. Sin embargo, Oculus Go tiene unas especificaciones inferiores y la capacidad de seguimiento del casco y los mandos no es tan buena.

Meta Quest 2, además, gracias a sus cámaras de seguimiento internas, permite prescindir de los mandos por completo y usar únicamente las propias manos del jugador para interactuar con el mundo juego. Ya que el dispositivo será utilizado por personas mayores, es necesario hacer la experiencia lo más sencilla posible en el lado técnico, por lo que la posibilidad de disminuir la dificultad de adaptación al uso de un mando, así como un seguimiento excelente permiten evitar problemas y confusión en las personas.



Figura 4.1: Meta Quest 2. [35]

Por estos motivos, se define Meta Quest 2 como plataforma ideal para este desarrollo. Sin embargo, a fecha del comienzo de este proyecto, no se dispone de ningún dispositivo de este tipo y se desconoce si se podrá disponer de él. Por lo que se comienza este trabajo utilizando unas HTC Vive, el dispositivo desarrollado conjuntamente entre Valve y HTC. Estas gafas, aunque potentes, requieren estar conectadas permanentemente a un ordenador y usan estaciones externas para el seguimiento de los movimientos del jugador. Durante el desarrollo del proyecto se consiguió acceso a un set de Meta Quest 2, por lo que se cambia el dispositivo durante el desarrollo. A pesar de generar trabajo extra, este cambio permite trabajar con bibliotecas más modernas y ofrece un resultado mucho más adecuado a las características de este proyecto como se ha definido en el inicio de esta sección.

Para desarrollar el juego es necesario también un motor de videojuegos, en este caso, tanto Unreal Engine 4 como Unity son compatibles con Meta Quest 2 y HTC Vive. En este caso se va a elegir utilizar Unity (figura 4.2) ya que tiene gran cantidad de documentación disponible sobre RV, está muy optimizado para la plataforma Android y además tiene paquetes y bibliotecas que permiten trabajar de manera más sencilla con Meta Quest 2 y HTC Vive.

La principal biblioteca que apoyará el desarrollo del proyecto es VRTK 4. Es

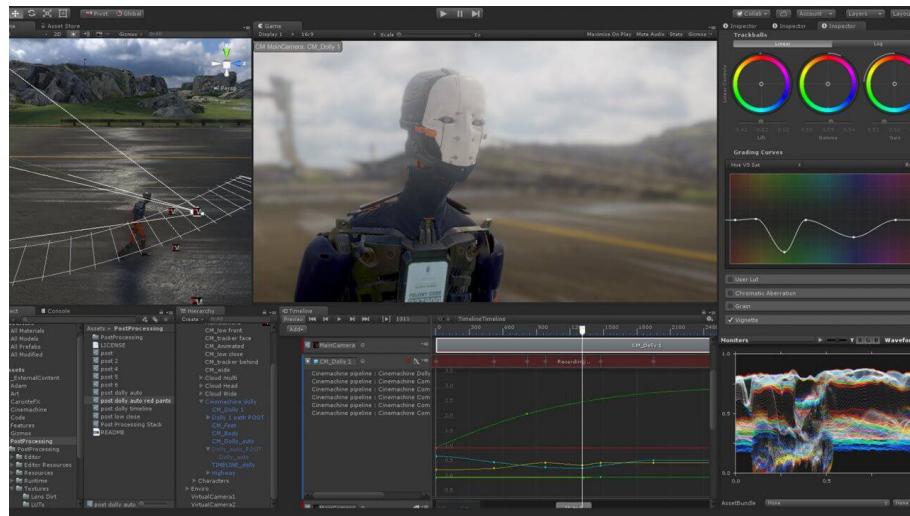


Figura 4.2: Interfaz de Unity. [26]

una compilación de diferentes scripts que permiten la fácil implementación de mecánicas indispensables en un videojuego de realidad virtual:

- Locomoción.
- Interacciones con objetos virtuales (coger, soltar, empujar...).
- Interacción con una interfaz dentro del espacio virtual.
- Controladores como botones y palancas.

Unity es un programa con un funcionamiento modular, lo que significa que pueden agregarse paquetes de funcionalidad de manera sencilla desde su tienda integrada: Asset Store (vease figura 4.3). Todos los paquetes necesarios para el desarrollo de este proyecto están disponibles en la Asset Store y se pueden descargar de forma directa desde Unity.

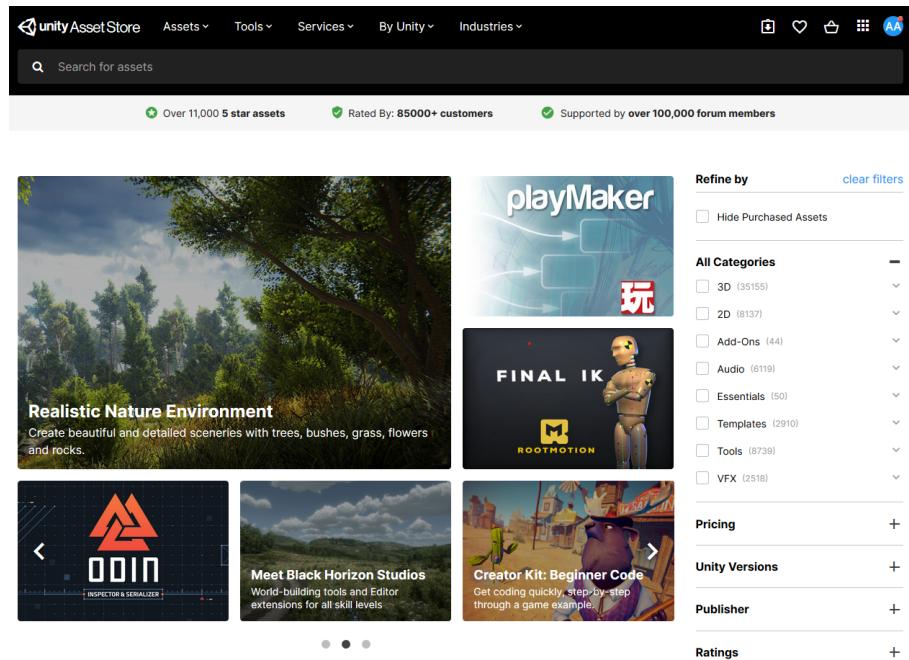


Figura 4.3: Portada de la Asset Store.

Adicionalmente, trabajar con VRTK permite cierta abstracción de la plataforma final del videojuego, ya que es compatible tanto con los dispositivos de Meta, como con todos aquellos cascos de RV compatibles con SteamVR, el sistema de RV de la empresa Valve. Esto permitirá que, una vez finalizado el proyecto, se pueda extender fácilmente para funcionar también con el resto de los dispositivos VR actuales y, posiblemente, futuros.

Capítulo 5

Metodología a usar

Hablar sobre git

5.1. Desarrollo y Metodología Ágil

El desarrollo de software es un proceso cuyo objetivo es la creación de un programa informático para la resolución de un problema o para obtener una funcionalidad deseada. Este proceso es muy extenso y suele incluir varias etapas, incluyendo, pero no limitándose a:

- **Estudio del problema.** Dónde se estudia y analiza la tarea en cuestión, detallando la solución deseada.
- **Detalle de las especificaciones.** Es necesario descomponer el problema para describir todas las posibles restricciones y condiciones de funcionamiento.
- **Diseño.** En esta etapa se crea una representación de qué forma va a tener el programa informático a desarrollar teniendo en cuenta las especificaciones.
- **Programación.** En este momento es cuando se creará el código específico del programa siguiendo el diseño anterior.
- **Documentación.** Es el proceso de crear la información que acompaña al programa para describir su funcionamiento y ayudar a su mantenimiento, tanto a nivel de usuario como a nivel de desarrollador.

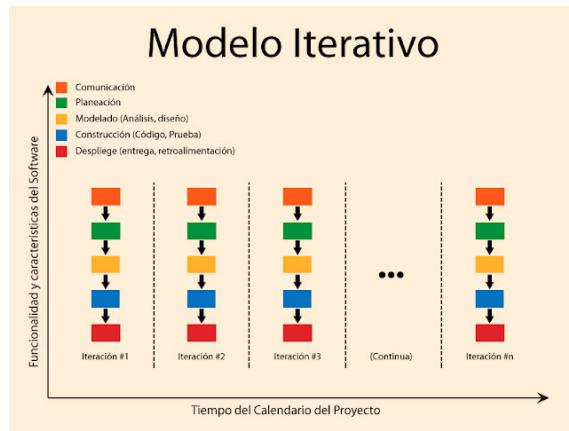


Figura 5.1: Diagrama de desarrollo iterativo. [36]

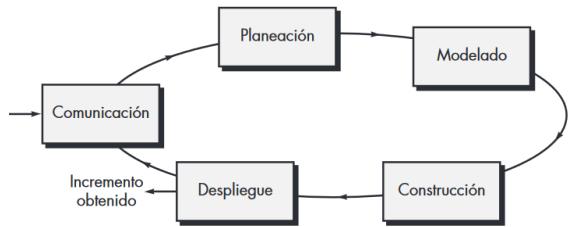


Figura 5.2: Diagrama de desarrollo evolutivo. [37]

- **Testeo.** Fase de pruebas en la que se comprueba que el programa funciona como es debido, no tiene errores y cumple con las especificaciones.
- **Mantenimiento.** Fase final del desarrollo en la que se da soporte a la aplicación, manteniéndola actualizada y arreglando fallos no detectados anteriormente.

Estas etapas están claramente diferenciadas y en el desarrollo clásico se siguen de forma secuencial, pero en la actualidad se siguen las llamadas metodologías ágiles, que permiten un desarrollo mucho más rápido, detectando errores antes y obteniendo un producto para el cliente en menos tiempo. Es común utilizar un modelo iterativo (véase figura 5.1), en el que estas fases se repiten tras obtener una retroalimentación del cliente, en lugar de dar fin al desarrollo.

El proceso evolutivo es similar y en ocasiones complementario al iterativo, ya que al final de cada iteración, se obtiene parte del resultado final, sobre el que en iteraciones posteriores se irán añadiendo funcionalidades. (véase figura 5.2)

Tanto el modelo iterativo como el evolutivo permiten un desarrollo centrado en el usuario, ya que el cliente final proporciona información sobre los desarrollos ya

entregados que permiten mejorar constantemente. Además de dar seguridad al equipo desarrollador de que su próximo trabajo tiene una base sólida, evitando tener que rehacer trabajo en una fase muy avanzada al descubrir un problema.

Por estos motivos, para este proyecto se va a utilizar una metodología evolutiva basada en entregas, obteniendo al final de cada entrega una parte funcional del producto final. La organización de estas entregas, así como las especificaciones para cada una de ellas se describen en detalle en la sección **Plan de Entregas** (6.1) y todo su contenido en la parte **Desarrollo** (III).

Para el caso particular de desarrollo de videojuegos se siguen las mismas metodologías que para cualquier otro software, pero adquiere especial importancia un documento llamado Game Design Document, donde se detalla toda la información del videojuego final.

5.2. Game Design Document

El Game Design Document o GDD es un documento en constante evolución que describe de la forma más detallada posible todos los aspectos de un videojuego. Al ser un documento específico para este tipo de desarrollos, permite alcanzar gran profundidad sobre todo de información del juego. Por norma general este documento se crea en colaboración entre todos los equipos de personas que trabajan en la creación del proyecto, desde programadores hasta artistas conceptuales o diseñadores de jugabilidad. El documento suele crearse antes de comenzar el desarrollo pero está abierto a actualizarse y añadir nueva información durante el proceso de desarrollo.

La función principal de este documento es plasmar la visión que tienen los equipos del producto final y servir de guía a todos ellos para alcanzar el objetivo de la forma deseada. También puede tener la función de carta de presentación entre un equipo de desarrollo y una editorial cuando los primeros buscan financiación.

Para el desarrollo de este proyecto se ha creado un GDD simplificado y adaptado a las particularidades de este Trabajo de Fin de Grado. El documento se encuentra en el apéndice **Game Design Document** (A).

5.3. Evaluación

Una parte importante para este proyecto es evaluar el juego creado y su usabilidad obteniendo datos a partir de pruebas con usuarios reales. La usabilidad es un aspecto fundamental en el desarrollo de software y habitualmente es definida como una métrica usada para evaluar la facilidad de uso del producto, su efectividad, eficiencia y la satisfacción que produce en los usuarios. [76]

Existen varios métodos para medir la usabilidad de un sistema, los heurísticos y los empíricos. Los primeros se basan en conocimientos teóricos para analizar detalladamente el funcionamiento de una aplicación. Por otra parte, los métodos experimentales se centran en obtener información de usuarios del programa. La forma más común es a través de cuestionarios de satisfacción o SUS (System Usability Scale).

Los cuestionarios SUS son usados en diversos ámbitos a parte del desarrollo software y son un estándar que permite evaluar la eficacia, eficiencia y satisfacción que proporciona cualquier sistema. Estos cuestionarios cuentan con diez enunciados cuidadosamente diseñados a los que se responde con un número del 1 al 5 (1 - Totalmente en desacuerdo y 5 - Totalmente de acuerdo). De esta forma se puede aplicar una sencilla fórmula matemática para obtener un resultado numérico indicando el grado de usabilidad del sistema. [38]

Los diez enunciados se dividen en 5 positivos y 5 negativos, intercalados entre sí. Para obtener la puntuación final hay que seguir estos pasos:

- Sumar los resultados de los enunciados positivos y restar 5.
- Restar a 25 la suma de los enunciados negativos.
- Por último, sumar los dos números anteriores y multiplicar el resultado por 2,5.

Como resultado se obtiene un número entre 0 y 100. Cuanto mayor sea el número, mejor será la usabilidad del sistema. Por norma general se entiende que un resultado aceptable debe ser superior a 68 puntos. (figura 5.3)

Se ha elegido el cuestionario SUS como método para realizar la evaluación de este proyecto principalmente por tratarse de una herramienta estándar en el sector, por su simplicidad para los usuarios que tienen que rellenarlo y por dar un resultado numérico fácilmente interpretable. El cuestionario utilizado junto con sus resultados se encuentra en el apéndice **Cuestionarios (B)**

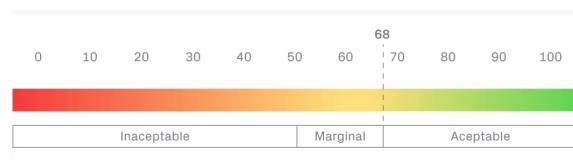


Figura 5.3: Representación de los resultados de un SUS. [38]

Capítulo 6

Plan de entregas

6.1. Entregas

El desarrollo de este proyecto estará dividido en una serie de entregas incrementales en las que se buscará crear un proyecto base ejecutable en un entorno de realidad virtual, al cual se le añadirán las distintas mecánicas del juego de manera gradual, obteniendo como resultado final tras la última entrega un videojuego completamente funcional y que cumpla los objetivos planteados para este trabajo de fin de grado.

- **Entrega 1:** Tener una escena en realidad virtual en la que un jugador pueda interactuar con el entorno usando avatares que representen sus manos.
 - Iteración 1: Crear un entorno funcional que sea capaz de ejecutarse de manera correcta en el dispositivo de VR de desarrollo.
 - Iteración 2: Incorporar avatares de manos para el jugador. Añadir las posibles interacciones del jugador con sus manos: coger, soltar y lanzar objetos e intercambiar un objeto de una mano a la otra.
 - Fecha de entrega estimada: 12 de mayo de 2023
- **Entrega 2:** Diseñar e integrar en el proyecto las mecánicas y métodos necesarios para crear las pruebas de las que constará el juego.
 - Iteración 1: Diseñar las técnicas básicas que se usarán para evaluar las pruebas.
 - Iteración 2: Crear en Unity las mecánicas básicas necesarias para realizar las distintas pruebas.

- Iteración 3: Poner a prueba la adaptación de varias personas al uso de la realidad virtual en general y en concreto a las mecánicas desarrolladas.
 - Fecha de entrega estimada: 24 de junio de 2023
- **Entrega 3:** Diseñar e incluir en el proyecto todos los escenarios y entornos necesarios para el juego.
 - Iteración 1: Diseñar los escenarios en los que se desarrolla en juego.
 - Iteración 2: Creación e integración de los distintos escenarios y decorados para el proyecto.
 - Iteración 3: Conocer la respuesta de varios usuarios ante los escenarios creados.
 - Fecha de entrega estimada: 8 de julio de 2023
- **Entrega 4:** Diseñar varias pruebas concretas de cada uno de los tipos y posteriormente añadirlas al proyecto.
 - Iteración 1: Diseñar todas las pruebas que habrá en el proyecto.
 - Iteración 2: Implementación de 3 pruebas de cada tipo.
 - Iteración 3: Evaluar la realización de las distintas pruebas por varios usuarios.
 - Fecha de entrega estimada: 29 de julio de 2023
- **Entrega 5:** Integrar todos los componentes para crear el flujo de juego y finalizar el proyecto.
 - Iteración 1: Unificar todo el desarrollo para crear la versión final jugable del proyecto.
 - Iteración 2: Probar y evaluar las experiencias de varios jugadores en la versión completa del juego.
 - Fecha de entrega estimada: 30 de agosto de 2023

Parte III

Desarrollo

Capítulo 7

Desarrollo del proyecto

Contenido

7.1. Entrega 1	64
7.1.1. Objetivo	64
7.1.2. Iteración 1	64
7.1.3. Iteración 2	67
7.1.4. Conclusiones de la entrega	71
7.2. Entrega 2	73
7.2.1. Objetivo	73
7.2.2. Iteración 1	73
7.2.3. Iteración 2	79
7.2.4. Iteración 3	87
7.2.5. Conclusiones de la entrega	90
7.3. Entrega 3	91
7.3.1. Objetivo	91
7.3.2. Iteración 1	91
7.3.3. Iteración 2	97
7.3.4. Iteración 3	103
7.3.5. Conclusiones de la entrega	104
7.4. Entrega 4	105
7.4.1. Objetivo	105
7.4.2. Iteración 1	105

7.4.3. Iteración 2	110
7.4.4. Iteración 3	118
7.4.5. Conclusiones de la entrega	119
7.5. Entrega 5	119
7.5.1. Objetivo	119
7.5.2. Iteración 1	119
7.5.3. Iteración 2	143
7.5.4. Conclusiones de la entrega 5	143

7.1. Entrega 1

7.1.1. Objetivo

El objetivo de esta entrega es tener un entorno de realidad virtual con el que el jugador pueda interactuar usando avatares que representen sus manos. El usuario deberá poder coger, soltar y lanzar objetos, así como intercambiar un objeto ya sujeto. El jugador debe tener visión libre de 360º, pero no es necesario que pueda moverse.

7.1.2. Iteración 1

Se busca crear un entorno funcional que sea capaz de ejecutarse de manera correcta en el dispositivo de VR de desarrollo (HTC Vive).

Se comienza creando un nuevo proyecto en Unity 2019.2.16f1. Una vez el proyecto se ha generado correctamente, es necesario importar los paquetes y bibliotecas necesarias para el proyecto:

- SteamVR Unity Plugin: El objetivo de este plugin es manejar la interacción entre Unity y SteamVR. SteamVR es una API que incluye soporte para varios dispositivos de RV y proporciona varias funcionalidades como la gestión de los botones en los mandos y la representación de estos en el mundo virtual.
- VRTK Prefabs: Es una colección de patrones para la computación espacial en realidad virtual.



Figura 7.1: Importación del paquete SteamVR Unity Plugin 2.6.0b1.

- Zinnia: Se trata de un conjunto de patrones de diseño con el objetivo de resolver problemas comunes en la RV.

El primer paquete en ser integrado es SteamVR Unity Plugin, para ello hay que acudir a su repositorio en GitHub (https://github.com/ValveSoftware/steamvr_unity_plugin) y descargar la versión más reciente (2.6.0b1). Una vez descargado el paquete, se importa a Unity mediante el menú dedicado a ello (figura 7.1). La integración se realiza correctamente, por lo que se pasa a la siguiente biblioteca.

Para importar VRTK Prefabs (1.1.11) basta con seguir la documentación disponible en su repositorio de GitHub (<https://github.com/ExtendRealityLtd/VRTK.Prefabs>). Solo es necesario modificar el archivo manifest.json del proyecto como se indica en la figura 7.2. Este archivo se encarga de gestionar las depen-



Figura 7.2: Líneas a añadir en manifest.json, siendo X.Y.Z la versión deseada.



Figura 7.3: Errores al importar VRTK Prefabs.

dencias de paquetes.

Integrar Zinnia (1.16.0) en el proyecto se realiza de la misma forma que VRTK Prefabs, modificando el archivo manifest.json según las instrucciones de su repositorio (<https://github.com/ExtendRealityLtd/Zinnia.Unity>).

En esta prueba se han utilizado las versiones más recientes de cada uno de los paquetes, pero tras integrar VRTK Prefabs comienzan a aparecer errores de compatibilidad con SteamVR (figura 7.3) y resulta imposible continuar.

Tras varias pruebas de integración, se han encontrado las versiones de cada paquete que no interfieren entre ellos. Por tanto, en el desarrollo de este proyecto no se van a utilizar las versiones más recientes hasta la fecha de cada uno (mostradas anteriormente), si no que se van a utilizar las siguientes versiones:

- SteamVR Unity Plugin 2.2.0
- VRTK Prefabs 1.1.5

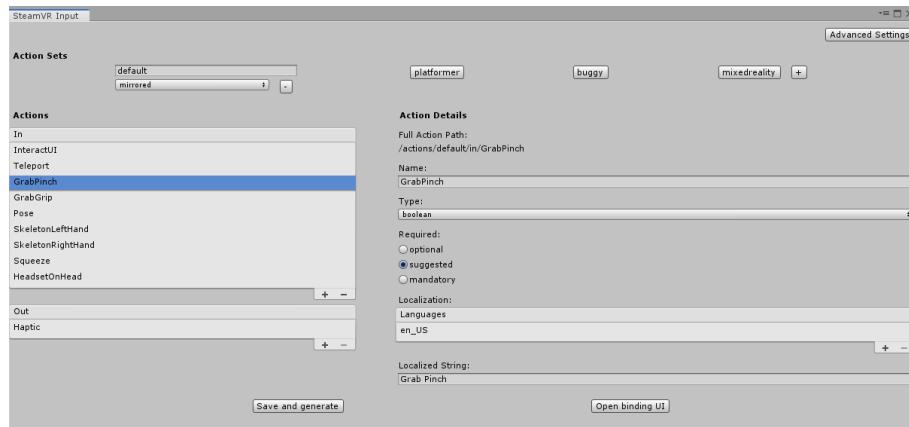


Figura 7.4: Ventana SteamVR Input.

■ Zinnia 1.8.0

Por último, es necesario generar los datos sobre las entradas de SteamVR, para ello se abre la ventana SteamVR Input (figura 7.4) y se pulsa el botón Save and generate.

Una vez todos los paquetes se han importado de manera correcta, se crea una escena en el proyecto de Unity y en ella se añade un prefab proporcionado por el plugin de SteamVR que contiene todo lo necesario para funcionar con el dispositivo de desarrollo.

7.1.3. Iteración 2

Es necesario incorporar un avatar que represente las manos del jugador, así como añadir las posibles interacciones que este pueda realizar: coger, soltar, lanzar y cambiar de mano.

El plugin de SteamVR ya proporciona un avatar para las manos del jugador, así como para los mandos (figura 7.5), por lo que de momento se utilizarán esos recursos.

La mayoría de las interacciones estarán gestionadas por VRTK, por lo que a continuación se va a crear la estructura que enlaza el objeto del jugador de SteamVR con los sistemas de VRTK. Esto permite una mayor diversidad de interacciones y compatibilidad con otros dispositivos de realidad virtual que no están soportados por SteamVR mediante la capa de abstracción de VRTK.

El primer paso es hacer que VRTK gestione el movimiento de la cámara y los mandos, para ello al objeto ‘Player’ de la escena se le añade y enlaza el

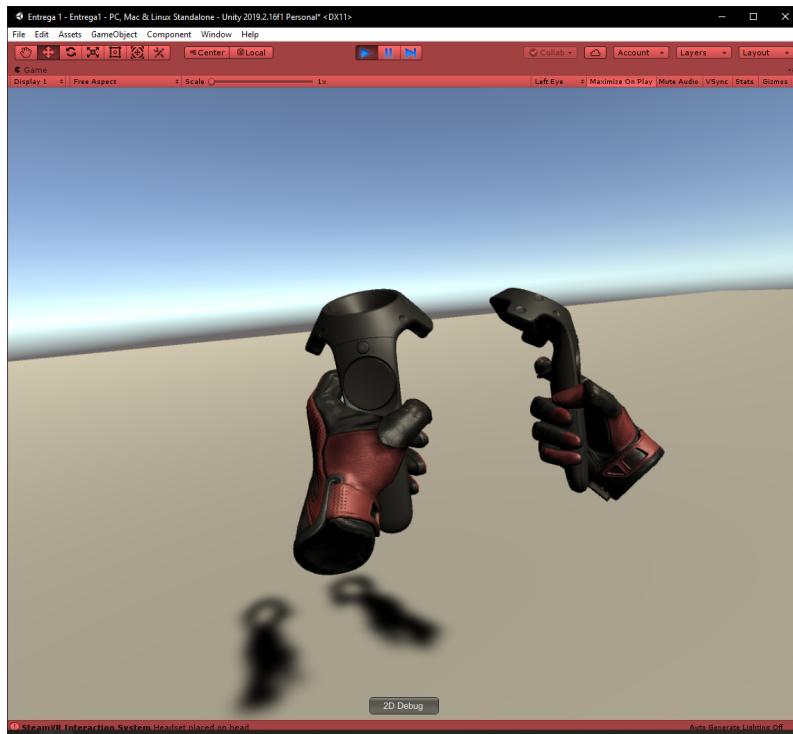


Figura 7.5: Avatar por defecto de SteamVR.

script de VRTK ‘Linked Alias Association Collection’ (figura 7.6). Este script es el encargado de recoger los datos de los mandos y el visor de RV y pasárselos a VRTK.

A continuación, se crea un sencillo script (figura 7.7) cuyo objetivo es obtener la velocidad de un mando de SteamVR y devolverla en un formato adecuado para VRTK. Este script se añade a los objetos que representan las manos del jugador y se enlazan con el script de la sección anterior.

Por último, se crea un objeto que sea capaz de recoger y utilizar los datos que el objeto ‘Player’ está enviando. Para ello se utiliza un prefab de VRTK: Tracke-dAlias. Este objeto puede recibir información de varios ‘Player’ provenientes de distintos paquetes de RV, en este caso, de SteamVR. Véase figura 7.8.

El segundo paso consiste en hacer que las acciones del jugador en los mandos (pulsaciones de botones) sean trasladadas a VRTK, para ello se utiliza el gestor de acciones nativo de Unity junto con un nuevo script (figura 7.9) que recoge una acción de SteamVR y activa o desactiva una acción de Unity.

Este nuevo script debe estar asociado a un objeto del juego, por lo que hay que crear dos GameObject vacíos que contendrán los manejadores con el script

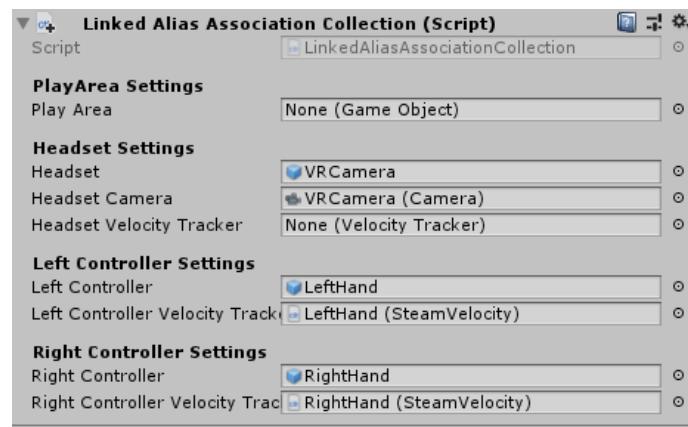


Figura 7.6: Script 'Linked Alias Association Collection' enlazado con los objetos que representan las manos y la cabeza.

```

1  using UnityEngine;
2  using Valve.VR.InteractionSystem;
3  using Zinnia.Tracking.Velocity;
4
5  O referencias
6  public class SteamVelocity : VelocityTracker
7  {
8
9      [Tooltip("The hand GameObject used by SteamVR")]
10     public Hand trackedGameObject;
11
12     O referencias
13     protected override Vector3 DoGetAngularVelocity()
14     {
15         return trackedGameObject.GetTrackedObjectAngularVelocity();
16     }
17
18     O referencias
19     protected override Vector3 DoGetVelocity()
20     {
21         return trackedGameObject.GetTrackedObjectVelocity();
22     }
23 }
```

Figura 7.7: Script para obtener las velocidades de un objeto de SteamVR.

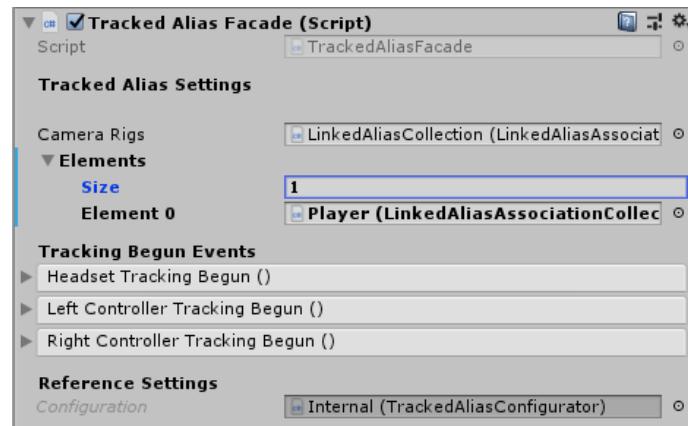


Figura 7.8: TrackedAlias enlazado con el objeto Player.

```

1  //using Valve.VR;
2  //using Valve.VR.InteractionSystem;
3  //using Zinnia.Action;
4
5  [Referencias]
6  public class SteamInputMap : BooleanAction
7  {
8      //action variable for input to VRTK
9      public string SteamVRAction;
10     public Hand hand;
11
12     private SteamVR_Action_Boolean button;
13
14     [Referencias]
15     private void Start()
16     {
17         button = SteamVR_Input.GetAction<SteamVR_Action_Boolean>(SteamVRAction);
18     }
19
20     [Referencias]
21     protected virtual void Update()
22     {
23         Receive(button.GetState(hand.handType));
24     }
25 }
```

Figura 7.9: Script para el manejo de acciones.

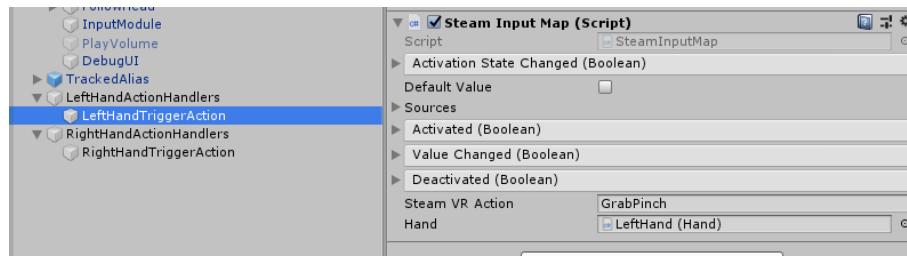


Figura 7.10: Manejadores para la acción de agarrar, activada por el gatillo del mando.

(figura 7.10), los cuales gestionarán las acciones generadas por cada uno de los dos mandos.

El último paso es añadir el objeto que proporciona VRTK para gestionar las interacciones de los mandos cuando utilizan las acciones manejadas en el paso anterior. Para ello se utiliza el prefab ‘Interactor’ y se anida dentro del objeto TrackedAlias, en concreto dentro de los alias de cada mando. Véase figura 7.11.

Este ‘Interactor’ tiene un script llamado ‘Interactor Facade’ con un parámetro que permite definir cual será la acción de agarrar un objeto. Así que se utiliza el manejador creado anteriormente (figura 7.12).

Finalmente es necesario añadir algún objeto con el que el jugador pueda interactuar. Para ello basta con crear un objeto 3D en la escena y añadirle los siguientes componentes (figura 7.13):

- RigidBody: permite que el objeto actúe regido por las físicas del juego.
- Interactable: Script que permite que un objeto sea interactivo en RV.

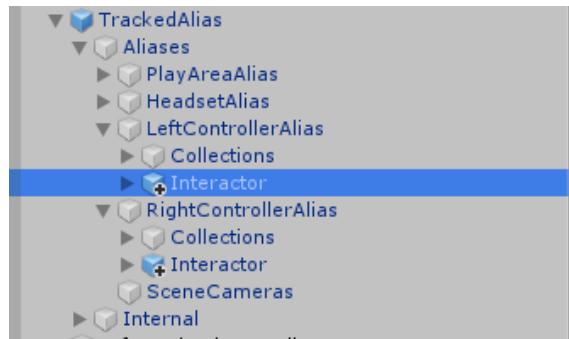


Figura 7.11: Interactors anidados en el objeto TrackedAlias.

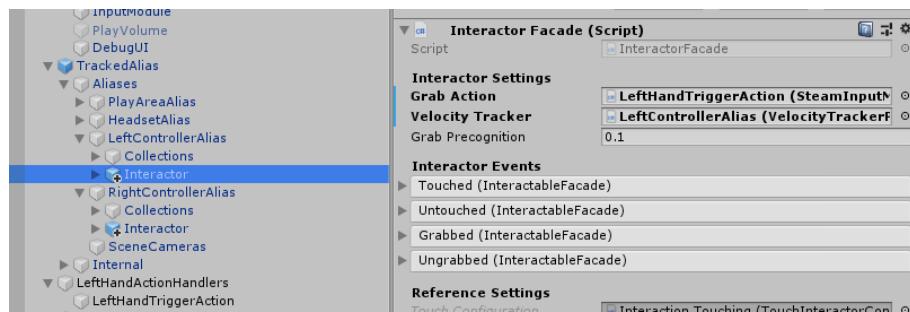


Figura 7.12: Interactor Facade con el manejador de acción definido anteriormente.

- Velocity Estimator: Puede estimar la velocidad de un objeto cuando el jugador lo suelta.
- Throwables: Hace que el jugador pueda lanzar el objeto.
- Steam VR Skeleton Poser: Permite definir la pose que adoptará la mano del jugador al interactuar con el objeto.

Con estos pasos terminados, el proyecto cuenta ya con un entorno en realidad virtual con todas las funcionalidades de SteamVR y VRTK. En este entorno el jugador puede coger objetos con sus manos, soltarlos, lanzarlos o cambiarlos de mano fácilmente (figura 7.14).

7.1.4. Conclusiones de la entrega

La versión utilizada de Unity permite el uso de RV mediante múltiples integraciones o plugins. Pero al ser un campo en pleno desarrollo no existe una marco estándar en el que todos los dispositivos de RV funcionen y sean compatibles entre sí. Además, esta evolución constante implica que las bibliotecas

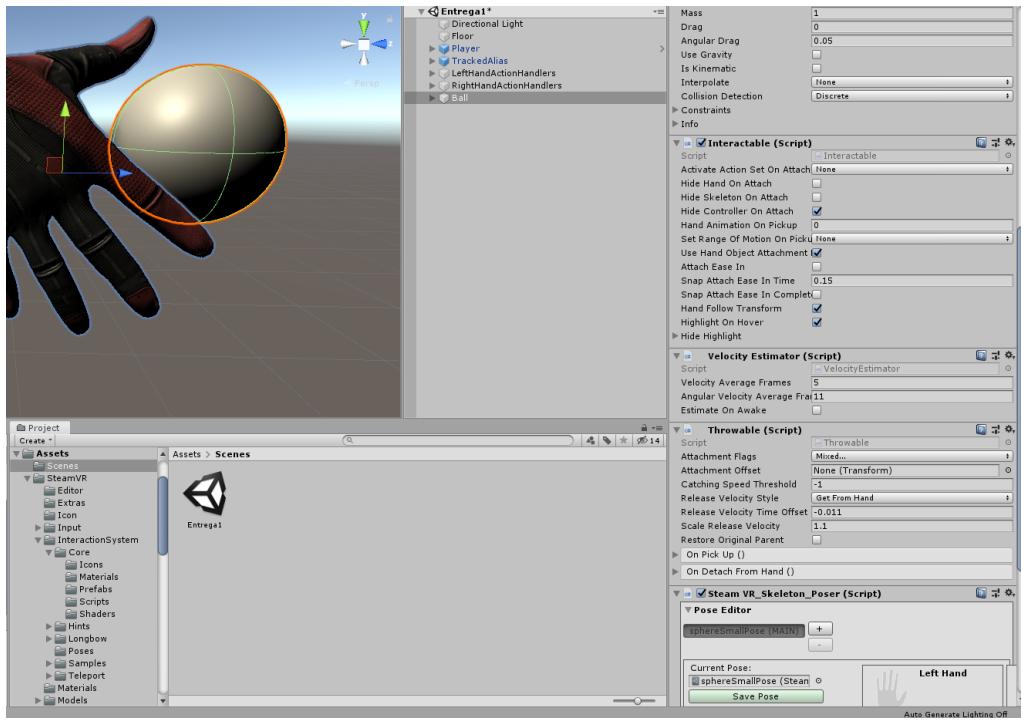


Figura 7.13: Objeto interactivo con los componentes necesarios para la RV.

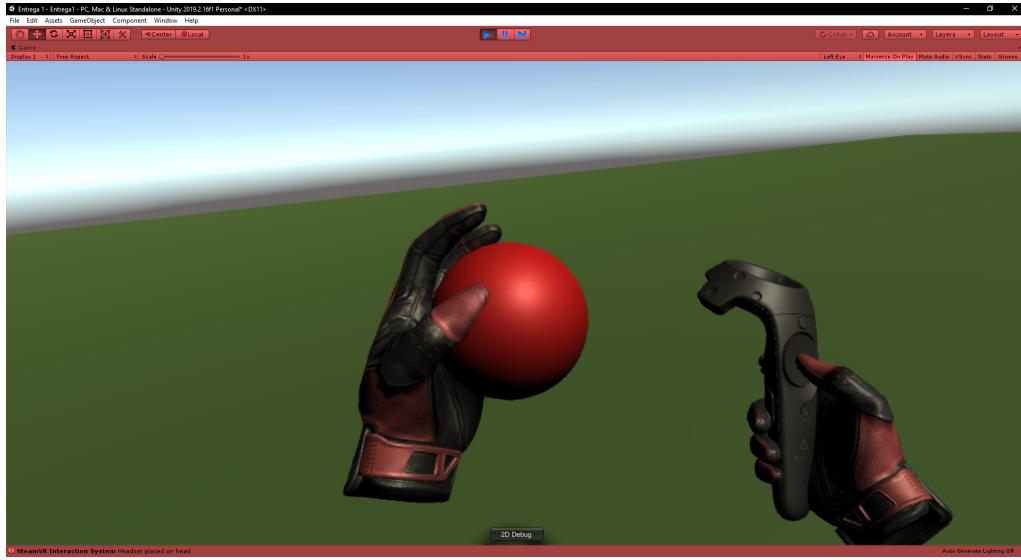


Figura 7.14: Avatar cogiendo una esfera interactiva.

actuales tengan cambios importantes entre versiones, dificultando encontrar la versión adecuada que funcione e implemente las capacidades requeridas para el dispositivo de RV en cuestión.

Esto implica que es posible que dentro de poco tiempo, cuando avance la tecnología y finalmente converja en un estándar, la mayoría de estas bibliotecas y plugins dejaran de funcionar y requerirán que los desarrolladores actualicen todas sus aplicaciones.

7.2. Entrega 2

7.2.1. Objetivo

El objetivo de esta entrega es diseñar e integrar en el proyecto las mecánicas y métodos necesarios para crear las pruebas de las que constará el juego. Estos elementos son los que permitirán detectar y medir los movimientos del jugador para juzgar que la prueba se ha hecho de manera correcta.

7.2.2. Iteración 1

Se busca diseñar las técnicas básicas que se usarán para evaluar las pruebas. En concreto encontrar las herramientas que ofrecen Unity y VRTK y cual es la mejor forma de utilizarlas para detectar que los movimientos y las acciones del jugador son las adecuadas para cada una de las pruebas.

Pruebas de posición

Hay varias pruebas que dependen exclusivamente de la posición de los mandos en el espacio virtual, por lo que es esencial poder hacer un seguimiento de su posición en todo momento. Asignando una serie de posiciones para cada controlador durante la prueba y conociendo su posición real se podría determinar si el jugador está realizando el movimiento adecuado.

Unity trabaja con un sistema de coordenadas cartesianas y todos los objetos del juego tienen un atributo que indica su posición en dicho sistema (figura 7.15), por lo que es sencillo obtener este dato.

Por desgracia, no se puede basar el sistema únicamente en la posición exacta del mando, es necesario dar ciertos márgenes que ayuden al jugador a mantener el mando dentro de una zona que será tomada como válida. Una opción sería

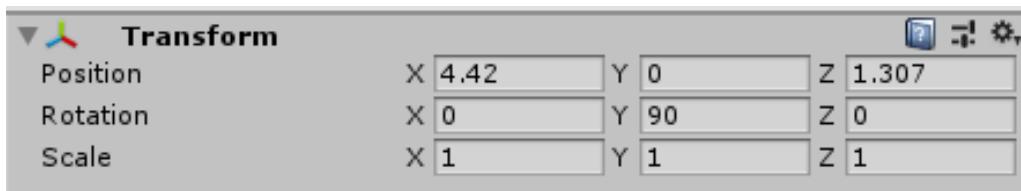


Figura 7.15: Ejemplo del atributo Transform en Unity que representa la posición de un objeto en el espacio.

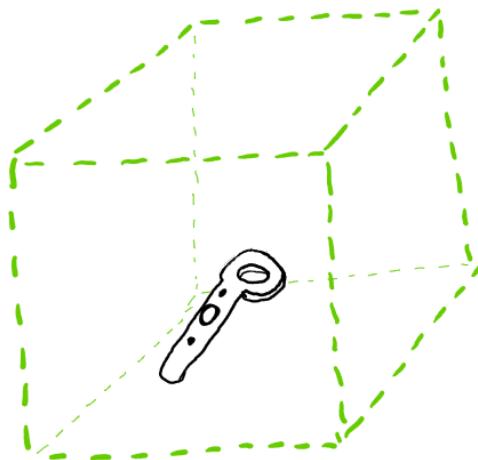


Figura 7.16: Boceto que muestra el uso de un trigger (cubo verde) para detectar el controlador en su interior.

definir un punto del espacio y una distancia máxima a la que el mando se puede encontrar para considerarse una posición correcta. Unity incorpora de forma nativa un objeto llamado ‘trigger’. Un trigger (representado en la figura 7.15) se trata de un objeto en el espacio virtual, de cualquier forma o tamaño, que es completamente invisible al jugador y es capaz de detectar cuando otro objeto entra dentro de él. De esta forma, se puede definir un trigger en cualquier posición que reportará al sistema cuando el mando entra o sale de su perímetro.

Teniendo la capacidad que ofrecen los trigger de Unity, definir las pruebas que se basan en la posición de las manos consiste en poder definir una serie de posiciones y tiempos en los que aparecerán dichos trigger para comprobar si el mando del jugador se encuentra en su interior. Puesto que los trigger son tratados como objetos comunes dentro de Unity, se pueden mover siguiendo una trayectoria y una velocidad, por lo que no solo pueden detectar posiciones, si no también evaluar el movimiento de los mandos entre dos puntos del espacio.

De esta forma ya se pueden definir algunas pruebas de movimiento:

Posiciones. En esta prueba se pide al jugador que adopte una pose concreta

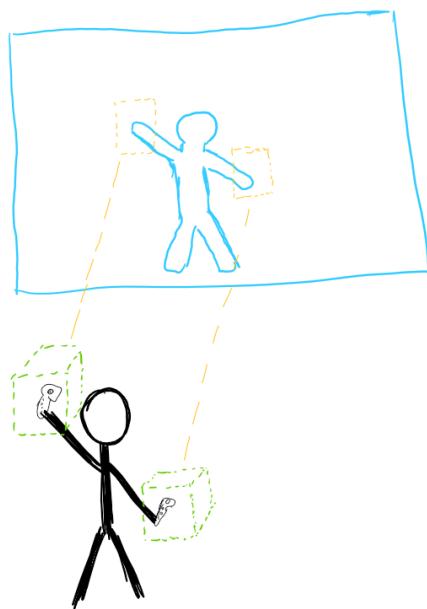


Figura 7.17: Boceto que representa una posible prueba de figuras. En azul el modelo a seguir, en verde los trigger necesarios.

que aparece representada (figura 7.17). Para este caso, para cada figura a imitar, es necesario definir una posición para cada mano, en dicha posición aparecerá un trigger. Si el jugador es capaz de colocar sus dos manos dentro de los trigger y mantenerlas en ellos, la prueba se dará como superada.

Baile/movimientos. (véase figura 7.18) En este caso se puede definir un movimiento como la trayectoria que sigue un trigger dentro del espacio. Para ello es necesario definir un punto por cada momento intermedio, así como la velocidad a la que el trigger se moverá entre ellos. Si el jugador es capaz de mantener la mano dentro del espacio designado durante todo el movimiento la prueba habrá sido superada. En el caso de la prueba de baile, esta consiste en un conjunto de movimientos y posiciones uno tras otro.

Objetivos. Esta prueba hace que el jugador vea como varios objetos se aproximan hacia él y debe tocarlos con la mano conforme llegan a su posición, según se observa en la figura 7.19. Cada objeto puede tener dos colores, el cual indica al jugador con qué mano debe tocarlo. Los objetos de color rojo se acercarán al jugador por su lado izquierdo y deberá tocarlos con su mano izquierda. Los objetos azules actuarán de la misma forma, pero para el lado derecho. Cada uno de estos objetos tendrá un ‘collider’ que es equivalente a un trigger, pero para objetos visibles al jugador, esto permitirá determinar si el jugador ha realizado la prueba de forma correcta.

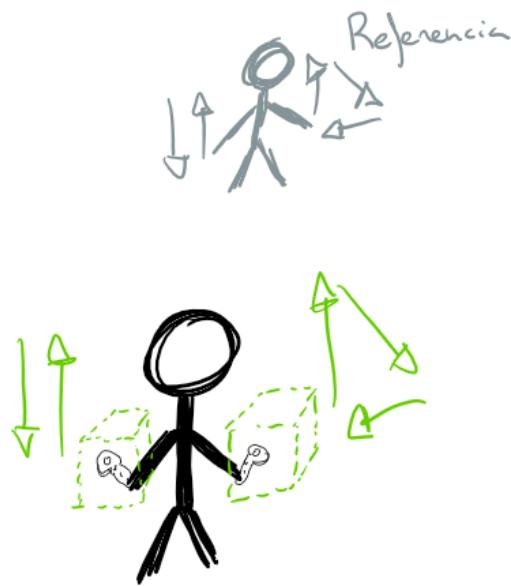


Figura 7.18: Boceto representando una prueba de baile o movimientos. En gris la referencia a seguir, en verde los trigger y los movimientos que siguen.

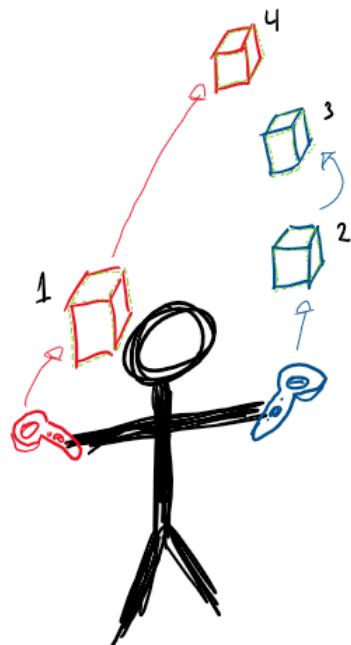


Figura 7.19: Boceto de una prueba de objetivos. Los objetivos en azul y rojo deberán tocarse con el mando del mismo color.

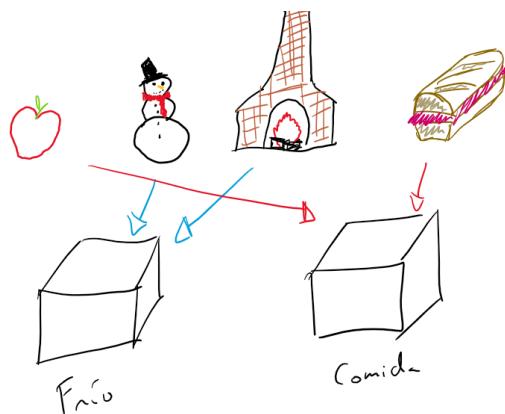


Figura 7.20: Boceto de la prueba de agrupación de objetos. Las categorías "frío comida" no son visibles para el jugador.

Pruebas de interacción

Algunas de las pruebas requerirán que el jugador interactúe con objetos virtuales, cogiéndolos, soltándolos o de formas similares. Por suerte, estas son capacidades básicas que ofrecen SteamVR y VRTK y que ya fueron integradas en el proyecto como parte de la primera entrega. Por lo que la definición de cada prueba consistirá en gran parte de la generación de los objetos virtuales: cuáles, dónde y cómo aparecen. Así como detectar lo que el jugador hace con ellos.

Para la prueba de agrupación de objetos (véase figura 7.20) se presentarán al jugador un grupo de objetos virtuales y esté deberá descubrir su asociación y colocarlos en dos zonas separadas. Estas zonas se pueden definir usando los trigger explicados anteriormente, de modo que basta con detectar cuándo todos los objetos han sido introducidos en alguna de las dos zonas, y evaluar si cada uno está donde le corresponde.

En la prueba de figuras superpuestas se mostrarán al jugador un conjunto de objetos virtuales o siluetas que se encuentran superpuestos unos con otros como se muestra en la figura 7.21. El jugador deberá averiguar de qué objetos de trata y responder seleccionando la respuesta correcta entre las disponibles que se presentarán mediante botones interactivos en RV. La funcionalidad de estos botones está proporcionada por los paquetes instalados en la entrega anterior.

Para la prueba de asociación de sonidos se presentan una serie de objetos al jugador de la misma forma que en la prueba de agrupación de objetos, pero en este caso el jugador deberá escuchar un sonido y decidir qué objeto es el que produce dicho sonido, como se ilustra en la figura 7.22. La única diferencia entre esta prueba y la de agrupación es que es necesario crear un objeto que sea capaz de reproducir un sonido previamente grabado y hacer que el jugador

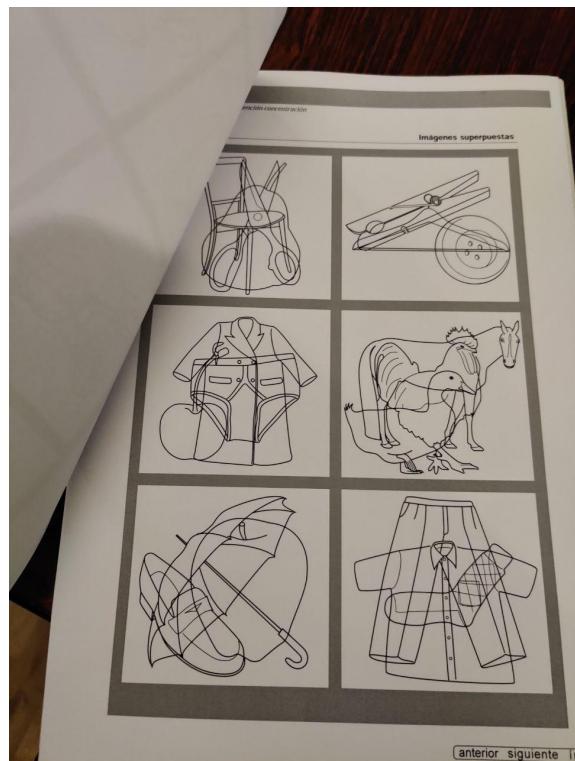


Figura 7.21: Fotografía de un ejercicio de figuras superpuestas en formato de papel.

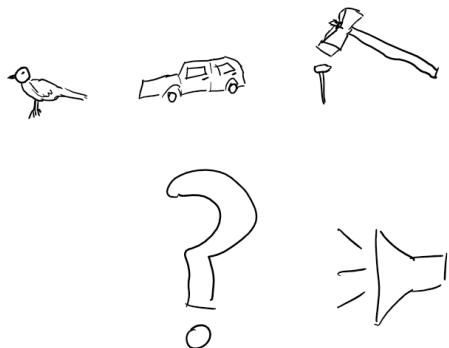


Figura 7.22: Boceto que representa la prueba de asociación de un sonido con el objeto que lo produce.

lo oiga. Por suerte esta es una funcionalidad básica de Unity.

Finalmente, para la prueba de localización de sonidos únicamente es necesario colocar un emisor de sonido en un punto del espacio rodeando al jugador. Unity es capaz de utilizar audio ambisónico, un tipo de sonido que permite al usuario localizar el punto exacto en el espacio de donde proviene dicho sonido. Por tanto, se usará esta tecnología para hacer que el jugador se gire en la dirección correcta de la que proviene el sonido. Cuando el jugador sea capaz de ver el objeto que producía el sonido, la prueba ha sido superada.

Pruebas de entorno

Estas son las pruebas en las que no se requiere que el jugador interactúe con el entorno de manera directa. Se presenta un entorno que el jugador debe observar y escuchar. Por tanto, para la evaluación de este tipo de pruebas es necesaria la participación de una persona externa al juego, preferiblemente una persona experta en entrenamiento cognitivo que actúe de guía para el usuario. Para estas pruebas no es necesario el uso de ninguna mecánica especial salvo la generación de los objetos virtuales del entorno.

7.2.3. Iteración 2

En esta iteración se busca crear en Unity las mecánicas básicas necesarias para realizar las distintas pruebas.

```
[Tooltip("Shows if a controller is inside the trigger")]
[SerializeField]
public bool controllerInside;

[Tooltip("Name of the tag used for the controllers")]
public string controllerTag;

[Tooltip("This material will be used when a controller is inside the trigger")]
public Material correctMaterial;
[Tooltip("This material will be used when a controller is outside the trigger")]
public Material incorrectMaterial;

[Tooltip("Number of controllers inside the trigger")]
[SerializeField]
private int nControllers;
```

Figura 7.23: Variables en el script para los triggers.

Pruebas de posición

Para implementar los triggers explicados en la iteración anterior se ha creado un objeto en Unity con forma de cubo y un tamaño que equivaldría a 50x50x50 centímetros en el mundo real. Durante el desarrollo del proyecto los triggers serán visibles para facilitar el trabajo, y serán de color rojo por defecto, pero se volverán verdes cuando detecten un mando en su interior, mostrando así su correcto funcionamiento.

Cada trigger contiene un único script que define su comportamiento durante el juego y es el encargado de detectar si hay un mando en su interior y reflejarlo de forma adecuada. El script utiliza las cinco variables mostradas en la figura 7.23, dos de ellas ('correctMaterial' e 'incorrectMaterial') son para almacenar los materiales que mostrará según haya o no algún mando en su interior. 'controllerTag' contiene el texto con el que se van a etiquetar los mandos, para de esta forma poder detectar si el objeto dentro del trigger es un mando. 'nControllers' almacena en todo momento el número de mandos que hay dentro del trigger, ya que es posible que haya varios en un mismo instante. Finalmente, 'controllerInside' contendrá un valor de verdadero o falso según haya algún mando en el trigger o no.

Durante el juego este script detecta la entrada y salida de cualquier objeto en el trigger mediante dos funciones nativas de Unity: `OnTriggerEnter` y `OnTriggerExit`, como se muestra en la figura 7.24. Las funciones se activan cuando se detecta un cualquier nuevo objeto, por lo que es necesario distinguir si se trata de un mando o no, para ello se utiliza la etiqueta "Controller" que está almacenada en la variable `controllerTag`. Si la etiqueta es la adecuada, quiere decir que un mando ha entrado o salido del trigger, por tanto, se actualiza el número de

```
0 referencias
private void OnTriggerEnter(Collider other)
{
    if(other.tag == controllerTag)
    {
        nControllers++;
    }
}

0 referencias
private void OnTriggerExit(Collider other)
{
    if (other.tag == controllerTag)
    {
        nControllers--;
    }
}
```

Figura 7.24: Métodos para manejar la entrada y salida de objetos en un trigger.

mandos que hay actualmente dentro del mismo.

Finalmente, el script comprobará de forma constante el número de mandos en su interior (figura 7.25), en caso de no haber ninguno, pondrá la variable ‘controllerInside’ a falso y usará el material rojo, indicando que no hay mandos en su interior. En caso contrario, el trigger usará el material ‘correctMaterial’ para mostrar que hay al menos un mando en él, así como activar la variable ‘controllerInside’. Por último, como medida de seguridad se comprueba que el número de mandos dentro del trigger no sea menor que 0 ni mayor que 2, y en caso de darse dicha situación, reasigna el valor a 0 si era menor que 0 o a 2 si era mayor que 2.

El siguiente problema planteado es la posición de los triggers en el espacio de juego. Los triggers deben estar en una posición fija con relación al jugador, pero debido a la naturaleza de la realidad virtual, el jugador puede moverse y rotar libremente, lo que podría causar que los triggers no se encuentren en el lugar esperado por el jugador. Para solucionar este problema se ha diseñado una matriz de posiciones posibles en las que puede aparecer un trigger (mostrada en la figura 7.26), y esta matriz se encuentra ligada a la posición del jugador y se mueve junto a él. De este modo los triggers siempre aparecen en el mismo lugar con respecto al jugador.

A continuación, es necesaria una forma de crear los triggers necesarios para cada nivel en su posición correcta, así como un método para expresar la posición

```
O referencias
private void Update()
{
    if (nControllers > 0)
    {
        GetComponent<MeshRenderer>().material = correctMaterial;
        controllerInside = true;
    }
    else
    {
        GetComponent<MeshRenderer>().material = incorrectMaterial;
        controllerInside = false;
    }

    nControllers = Mathf.Clamp(nControllers, 0, 2);
}
```

Figura 7.25: Parte del script que controla el número de mandos dentro de un trigger.

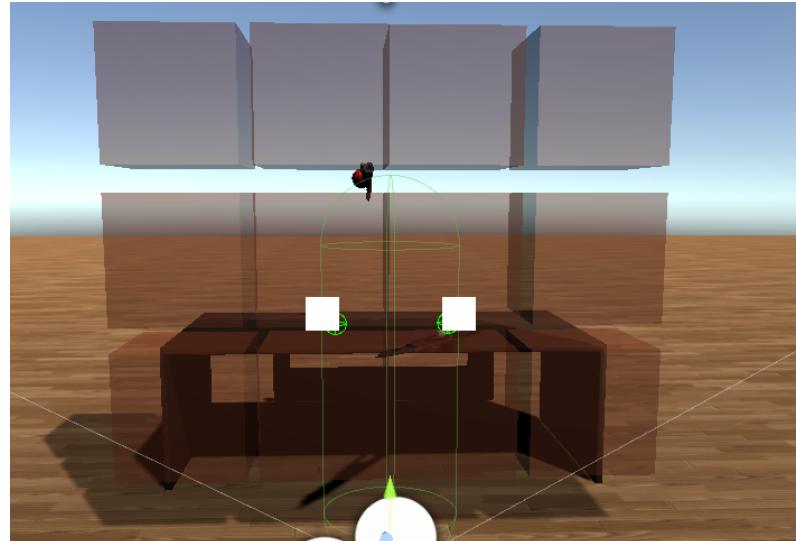


Figura 7.26: Representación de la matriz con 3 filas y 4 columnas de triggers. En el centro, una capsula verde representa al jugador, y los cubos blancos sus manos.

```
{  
    "level": {  
        "part": [  
            {  
                "time": 5,  
                "triggerPositions": [  
                    false, true, true, false,  
                    false, false, false, false,  
                    false, false, false, false  
                ]  
            },  
            {  
                "time": 10,  
                "triggerPositions": [  
                    false, false, false, false,  
                    true, false, false, true,  
                    false, false, false, false  
                ]  
            },  
            {  
                "time": 15,  
                "triggerPositions": [  
                    false, false, false, false,  
                    false, true, true, false,  
                    false, false, false, false  
                ]  
            }  
        ]  
    }  
}
```

Figura 7.27: Ejemplo de la estructura JSON representando un nivel con 3 partes.

de cada trigger para cada nivel. Como primera solución a este problema se ha optado por diseñar usando JSON una forma de expresar las posiciones de la matriz que deben estar ocupadas por un trigger, usando un objeto en Unity capaz de leer dicho formato desde un fichero y cargar los triggers necesarios en el entorno virtual.

El formato elegido para la representación de un nivel en JSON se muestra en la figura 7.27 y está compuesto por un campo “level” que encapsula todos los datos y contiene una lista de secciones llamada “part”. Cada sección corresponde a un estado concreto de la matriz de triggers, y cada uno de esos estados está definido por un “time” que indica cuantos segundos tardará en activarse dicho estado desde el comienzo de la prueba; y una matriz “triggerPositions” que contiene valores true/false dónde un ‘true’ significa que el trigger que ocupa dicha posición de la matriz estará activo y un false significa que no habrá ningún trigger en dicha posición.

En el ejemplo presentado en la figura 13 se muestra un nivel que empieza sin ningún trigger activo, a los 5 segundos del comienzo de la prueba se crean los dos triggers centrales de la primera fila. Pasados 10 segundos desde el comienzo, se desactivan los triggers anteriores y se activan los dos más exteriores de la

fila central. Cinco segundos después se cargan los triggers finales, los centrales de la segunda fila.

Para las pruebas que requieren el movimiento de los mandos de un punto a otro, se puede crear un trigger aislado que siga una trayectoria a una velocidad determinada, pero teniendo este sistema de gestión de los triggers mediante una matriz y la capacidad de cargar mediante un fichero diferentes configuraciones de la misma que se modifiquen en momentos determinados, se puede modificar el procedimiento para las pruebas de movimiento de forma que se adapten a este sistema y faciliten el desarrollo del proyecto.

Con el primer diseño es necesario definir un punto de inicio del trigger (con los problemas respecto a su posición relativa explicados anteriormente), así como una serie de puntos intermedios junto con un tiempo que debe tardar el trigger en alcanzarlos. Usando el diseño desarrollado para los triggers estáticos se logra resolver tanto los problemas de posición relativa, mediante el uso de la matriz que sigue al jugador, como el problema del tiempo entre cada punto intermedio gracias a que se puede definir en el fichero el momento en el que aparecerá cada trigger. El único impedimento restante es contabilizar de manera correcta el movimiento del usuario de un trigger al siguiente con una velocidad adecuada. Para ello, el objeto controlador de la prueba se encarga de contar el tiempo que el jugador está fuera del trigger entre la primer y la segunda posición, de esta forma, se puede averiguar si el movimiento del jugador ha sido fluido y acorde a lo esperado.

Esta solución para las pruebas de movimiento es menos precisa que la original, pero permite una mayor simplicidad y modularidad a la hora de construir niveles ya que se basa en el sistema desarrollado anteriormente.

Pruebas de interacción

Para este tipo de pruebas se diseñó en primer lugar utilizar los mismos triggers que se han usado hasta el momento, pero durante la implementación se ha encontrado una funcionalidad de VRTK que se ajusta más al resultado deseado: 'InteractableSnapZone'. Estos son objetos que forman una zona en la que si se coloca un objeto, cuando se suelte dicho objeto este se moverá automáticamente para colocarse en una posición y orientación determinada (figura 7.28) de forma que el objeto queda siempre colocado de la misma forma. Además, ofrece esta funcionalidad con un rango de acción variable, es decir, la distancia desde la cual será capaz de colocar un objeto en el lugar adecuado.

En la prueba de agrupación de objetos, que consiste en que al jugador se le presentan una serie de objetos y tiene que clasificarlos en dos grupos moviéndo-

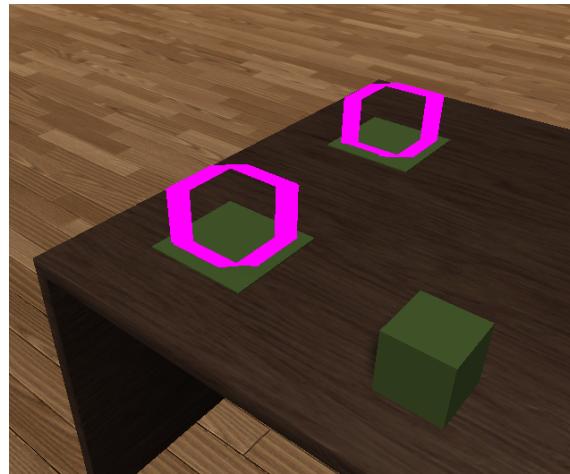


Figura 7.28: Ejemplo que muestra dos plataformas verdes con dos InteractableSnapZone. En rosa se remarcá la posición que ocupará el cubo verde si se suelta dentro de una de dichas zonas.

los y colocándolos en sus zonas correspondientes, se usan dos zonas de snap y mediante un script se puede comprobar qué objetos hay en cada zona.

En el script presentado en la figura 7.29 aparecen dos de las funciones utilizadas para comprobar que cada snap zone contiene un objeto de la categoría que se desea. El método CheckSnappedObjectsCorrect recorre la lista ‘list’ que contiene todos los snap zones del nivel y para cada zona que hay en el nivel comproueba que tiene dentro un objeto de una categoría válida mediante la función CheckSnappedTags. Esta función compara la etiqueta que describe a la zona de snap con la etiqueta que describe al objeto de su interior, y en caso de ser de la misma categoría (en el ejemplo: cat1 o cat2), devuelve un valor verdadero que indica al controlador que el objeto es válido. Si existe alguna zona que contenga un objeto no válido, se da la prueba por fallida, pero en caso contrario la prueba habrá sido superada.

Para las pruebas de objetos superpuestos y de asociación de sonidos se utiliza un botón que el jugador tendrá que pulsar para indicar la respuesta correcta. La funcionalidad básica de los botones viene dada por SteamVR y consiste en un objeto que el jugador puede mover en un eje empujándolo con la mano (figura 7.30). En este caso no es necesario agarrar el botón o interactuar de forma explícita con él, basta con que el jugador lo empuje para que el botón se active y ejecute un script, que es este caso indica si el botón pulsado es el correcto.

Finalmente, para la prueba de localización de sonidos se utiliza la capacidad que tiene Unity para generar audio 3D mediante el uso de su fuente de audio (figura 7.31) junto con el espacializador de Oculus que es necesario configurar como se muestra en la figura 7.32. De esta forma solo es necesario colocar una

```
0 referencias
public bool CheckSnappedObjectsCorrect()
{
    bool correct = true;
    for (int i = 0; i < list.Count && correct; i++)
    {
        correct = CheckSnappedTags(list[i].tag, list[i].SnappedGameObject.tag);
    }

    return correct;
}

1 referencia
public bool CheckSnappedTags(string snapZoneTag, string objectTag)
{
    bool correct = false;

    switch (snapZoneTag)
    {
        case "SnapZone_cat1":
            if (objectTag == "SnappedObject_cat1")
                correct = true;
            break;

        case "SnapZone_cat2":
            if (objectTag == "SnappedObject_cat2")
                correct = true;
            break;

        default:
            break;
    }

    return correct;
}
```

Figura 7.29: Dos métodos del script que controla los objetos dentro de cada snap zone.

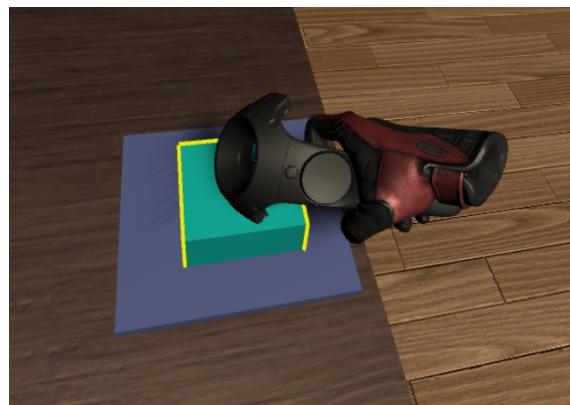


Figura 7.30: Jugador pulsando un botón.

fuente de sonido en cualquier punto alrededor del jugador y comprobar si está mirando hacia el origen del audio. Para hacer esta comprobación se usa una función del motor de renderizado de Unity que permite saber si un objeto va a ser dibujado en pantalla o no. Puesto que los objetos no visibles no se dibujan, de esta manera se determina si el jugador está viendo la fuente de audio mediante un sencillo script mostrado en la figura 7.33.

Con esto quedan implementadas en el proyecto todas las mecánicas básicas que son necesarias para comprobar si el jugador realiza las pruebas de manera correcta o no.

7.2.4. Iteración 3

En la última iteración de esta entrega se ha buscado poner a prueba la adaptación de varias personas al uso de la realidad virtual en general y en concreto a las mecánicas desarrolladas en esta entrega. Se ha hecho hincapié en la facilidad de uso y la comodidad del usuario durante el juego.

Para esto se ha desarrollado una pequeña escena de prueba en la que primero el jugador debe adoptar una postura corporal de forma que se activen de forma correcta unos triggers preparados previamente y que cambian cada 10 segundos hasta 3 veces. A continuación, el usuario se encuentra ante una mesa virtual (figura 7.34) en la que encuentra varios objetos, dos cubos verdes y dos cubos azules. En los bordes de la mesa hay 4 plataformas de los mismos colores que los cubos. El jugador puede interactuar libremente con estos objetos, pero debe colocar cada cubo en una plataforma de su color correspondiente. Una vez realizada la tarea, debe pulsar el botón situada en la parte derecha de la mesa. La pulsación de este botón provoca que se reproduzca un sonido proveniente de la parte trasera del usuario. Finalmente, el jugador tiene que girarse para ver la fuente del sonido y terminar así la prueba.

Esta prueba se ha realizado con varias personas mientras se monitorizaba mediante Unity todos los resultados y el funcionamiento de las mecánicas. Han participado tres personas que se caracterizan de la siguiente forma:

- Persona mayor de 50 años con poca experiencia tecnológica y ninguna experiencia en RV.
- Persona entre 20 y 25 años, con experiencia tecnológica, pero sin experiencia en RV.
- Persona entre 20 y 25 años con gran experiencia tanto en tecnología como en RV.

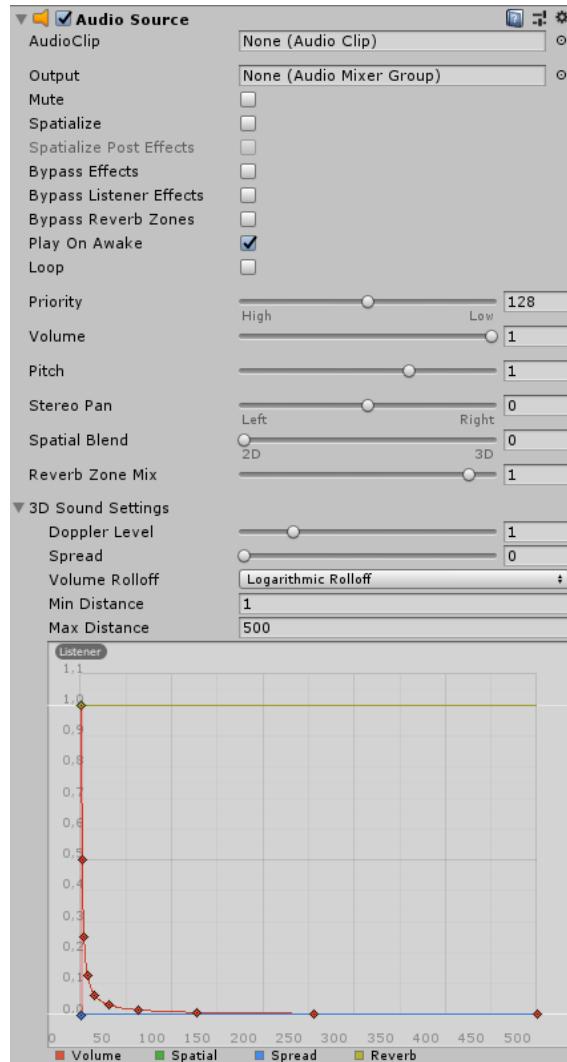


Figura 7.31: Muestra de la configuración de una fuente de audio en Unity.

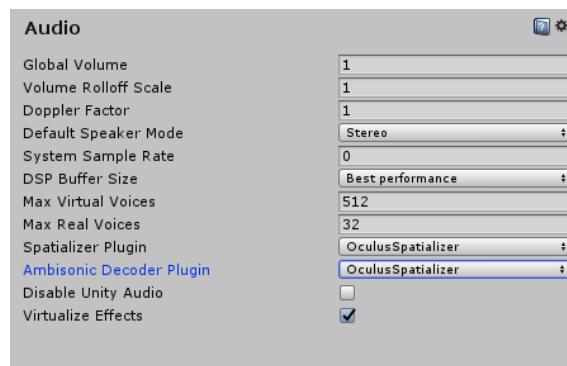


Figura 7.32: Ventana de configuración en la que es necesaria seleccionar OculusSpatializer para utilizar el sonido 3D.

```
O referencias
public bool Check AudioSourceVisible()
{
    bool visible = false;

    visible = this.GetComponent<Renderer>().isVisible;

    return visible;
}
```

Figura 7.33: Método que comprueba si la fuente de audio es visible por el jugador. Este script debe estar en la propia fuente.

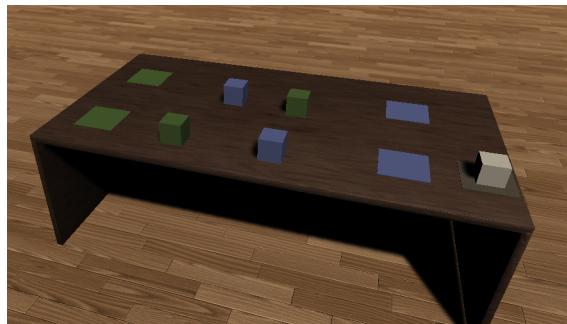


Figura 7.34: Mesa con objetos de prueba para las interacciones básicas, snap zones y botones.

Tras guiar a estas tres personas durante la prueba y posteriormente entrevisitarlas sobre su experiencia se han obtenido información útil para el desarrollo del proyecto y que se expone a continuación.

- Todas las personas han necesitado un breve periodo de adaptación a la realidad virtual, en el caso de las personas jóvenes ha sido más corto y ha consistido principalmente en ajustes físicos del casco de realidad virtual como la tensión de sus amarres o la distancia focal de las lentes para evitar ver de forma borrosa. En la persona mayor ha sido necesario más tiempo para adaptarse al entorno virtual, pero no provocando mareo ni otros efectos perjudiciales.
- Aunque ninguna persona ha tenido dificultades en seguir la prueba de posiciones, se ha observado que es necesario reajustar el tamaño y posición de los triggers, así como su cantidad y separación.
- En el uso de las manos virtuales las dos personas sin experiencia en RV prefieren que el avatar muestre una representación virtual del mando que están sosteniendo para facilitar la inmersión. Sin embargo, a la hora de interactuar y coger objetos han encontrado más difícil utilizar el avatar con mando por la discordancia con el mundo real en el que no se puede coger otro objeto mientras ya se tiene uno sujetado. La persona con experiencia en

RV ha preferido el avatar sin representación del mando en todo momento. Por tanto, se concluye que el mando virtual es útil para crear una inmersión inicial, pero debe ser sustituido cuando llega el momento de realizar interacciones con las manos.

- Durante la prueba de agrupación de los objetos por colores, las dos personas sin experiencia en RV han tenido ligeros inconvenientes a la hora de coger objetos principalmente por la estimación de la distancia a la que se encuentran. Estos inconvenientes no han sido graves y ambas personas se han acostumbrado rápidamente.
- Llegado el momento de intentar localizar la posición de la fuente de sonido a sus espaldas todo el mundo ha tenido problemas en discernir de dónde procedía el sonido. Tras investigar el problema es posible que por la naturaleza del audio digital y el dispositivo de reproducción usado sea difícil distinguir entre ciertas posiciones de una fuente como por ejemplo detrás del jugador o justo sobre su cabeza. Por lo tanto, esta prueba deberá ser acotada para que la fuente de sonido solo pueda aparecer en lugares en los que sea posible localizarla de forma cómoda.

7.2.5. Conclusiones de la entrega

Para la creación y diseño de las pruebas se han tenido en cuenta tres bases: ejercicios ya existentes y que se utilizan en entrenamiento cognitivo en la actualidad, interacciones sacadas de la ambientación deseada del juego (concurso de TV) y conceptos de juegos y experiencias de RV actuales que no tienen relación con el entrenamiento cognitivo. De esta forma se ha buscado adaptar ejercicios cognitivos conocidos al entorno de realidad virtual siguiendo sus paradigmas y tomando inspiración de juego actuales, y finalmente adaptando todo ello a la ambientación del juego en desarrollo.

Para la implementación en el proyecto de las pruebas diseñadas se ha intentado usar lo máximo posible las herramientas proporcionadas por Unity y VRTK y usando estándares de la programación y las estructuras de datos como JSON. De esta forma se busca hacer el juego adaptable y compatible con tecnologías futuras o de dispositivos de RV diferentes, manteniendo el proyecto y su código legible y posible de mantener por cualquier persona, así como favorecer su modificación. Por ejemplo, usando un formato JSON para determinar posiciones donde colocar las manos, permite prototipar rápidamente o incluso añadir nuevos niveles de forma fácil y sin necesidad de ninguna otra modificación al juego.

Finalmente, en cuanto a la prueba con personas de distintas edades y conocimientos tecnológicos, se han alcanzado las siguientes conclusiones:

- Todas las mecánicas diseñadas e implementadas han funcionado de manera correcta, aunque requieren de ajustes y cambios para funcionar de una forma más adecuada.
- El factor más importante de las personas era su experiencia anterior con la realidad virtual. Aunque todas las personas se han adaptado a la RV y han podido completar la prueba, aquellas sin experiencia han necesitado mayor tiempo de adaptación y han tenido más problemas en el manejo durante la prueba.
- El siguiente factor para tener en cuenta es la edad o experiencia tecnológica. La persona en el rango de más de 50 años y sin conocimientos tecnológicos ha necesitado más tiempo para adaptarse al espacio virtual, pero lo ha conseguido de igual forma y ha podido terminar la prueba de manera correcta.
- Es importante proporcionar al usuario un tiempo al principio del juego para acostumbrarse al entorno virtual, ya que en caso contrario podría provocar efectos no deseados en el jugador. Esto es especialmente importante para el público al que está dirigido este proyecto.

7.3. Entrega 3

7.3.1. Objetivo

El objetivo de esta entrega es diseñar e incluir en el proyecto todos los escenarios y entornos necesarios para el juego. El proyecto cuenta con un escenario principal donde transcurre la mayoría del juego, así como varios escenarios de menor tamaño y complejidad que se utilizarán en cada una de las distintas pruebas.

7.3.2. Iteración 1

En esta iteración se han diseñado todos los escenarios en los que se desarrolla el proyecto. El primer paso es decidir como se conectarán los escenarios entre sí.



Figura 7.35: Plató del programa de televisión 'Atrapa un millón'. [34]

La primera opción consiste en mantener cada escenario aislado del resto, haciendo que el jugador comience en el principal y sea transportado al escenario correspondiente a cada prueba cuando vaya a realizarla, siendo transportado de nuevo al escenario principal al completarla. Esta aproximación permite una mayor diferenciación y caracterización de cada escenario, pero puede resultar poco inmersiva ya que el jugador se ve transportado constantemente sin importar sus acciones.

La segunda opción mantiene todos los escenarios unidos entre sí en todo momento, haciendo que los escenarios de las pruebas formen parte del principal, esto aumenta enormemente la inmersión de usuario ya que todos los elementos del juego están a la vista y no hay movimientos de cámara ni posición ajenos al jugador. Con este enfoque se consigue más realismo y al mismo tiempo se reduce el tamaño y la carga de trabajo asociada a cada escenario, ya que las pruebas no necesitarán un entorno completo, si no que se reutiliza gran parte del escenario principal.

Finalmente, por los motivos ya expuestos, se decide utilizar la segunda opción y crear un escenario principal que contenga diferentes partes para cada prueba.

Este proyecto toma prestadas muchas ideas y conceptos de los concursos televisivos, por lo que son una fuente de inspiración ideal para la estética y el diseño de los escenarios (véanse figuras 7.36 y 7.35).

Como se muestra en las figuras 7.37 y 7.38, el escenario principal será circular, con el jugador situado en el centro. En los alrededores del habrá gradas con público (coloreadas de azul en las figuras), excepto en un trozo en el que se



Figura 7.36: Plató del programa de televisión '¡Ahora caigo!'. [39]

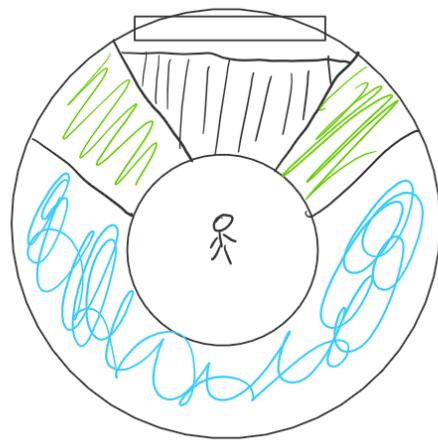


Figura 7.37: Boceto de la vista superior del escenario principal. En azul la zona de público, en verde la zona de atrezo, en negro la parte intercambiable para las pruebas.

situará una pantalla gigante en la que se presentará información al jugador (rectángulo superior). En este caso, se ha designado una zona del escenario que es intercambiable (rayada en negro en las figuras), y en dicha zona aparecerá el escenario adecuado para cada prueba subiendo desde el suelo hasta la posición del jugador de forma similar a algunos elevadores en escenarios de espectáculos. Finalmente, la zona verde en las figuras contendrá elementos típicos de un plató de televisión como focos y altavoces, y su objetivo es encuadrar la zona donde transcurren las pruebas y se da información importante al jugador.

Para la zona intercambiable del escenario que contendrá los distintos decorados, se ha buscado llenarla de objetos y atrezo relacionado con la prueba correspondiente, de forma que, al comenzar una prueba, aparecerá desde abajo todo el decorado y objetos necesarios para realizar dicha prueba. Tras finalizar

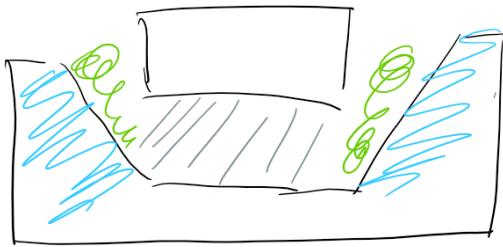


Figura 7.38: Boceto de la vista de perfil del escenario principal. En azul la zona de público, en verde la zona de atrezo, en negro la parte intercambiable para las pruebas.

la prueba, el escenario vuelve a su configuración original. Por tanto, solo es necesario diseñar un trozo pequeño del escenario para cada tipo de prueba.

Por lo general, las pruebas se pueden dividir en dos según el uso de su escenario: las que utilizan la pantalla principalmente y el resto es decoración, y las que utilizan los objetos virtuales del escenario como parte de la prueba y la pantalla sirve de decoración.

Pruebas de pantalla

Estas son las pruebas que utilizarán principalmente la pantalla y el resto del escenario contendrá objetos decorativos acordes con el tipo de prueba. La prueba de baile (figura 7.39) utiliza la pantalla para mostrar al jugador los movimientos que tiene que realizar en cada momento, en este caso el escenario contendrá objetos como instrumentos musicales, tocadiscos, o una persona bailando.

La prueba de turismo también es de este tipo. En la pantalla de muestra una imagen de un monumento o lugar famoso que el jugador debe descubrir mientras que el resto del escenario se llena con objetos referentes a los viajes, como se muestra en la figura 7.40.

En las pruebas de situaciones y descripciones se sigue el mismo diseño, en la pantalla se muestra una imagen o situación que el jugador debe comentar. Pero estas pruebas podrían ser del segundo tipo y usar el escenario en lugar de la pantalla si la situación a describir es una que se puede representar con objetos virtuales animados en lugar de una foto o video en la pantalla.

La prueba de adivinar la canción es un caso especial, ya que, tratándose de sonido, no es necesario el uso de la pantalla, pero de igual forma se puede decorar el escenario como se muestra en la figura 7.41.

Finalmente, en la prueba de superposición de objetos, se mostrará una ima-



Figura 7.39: Boceto del escenario para la prueba de baile.

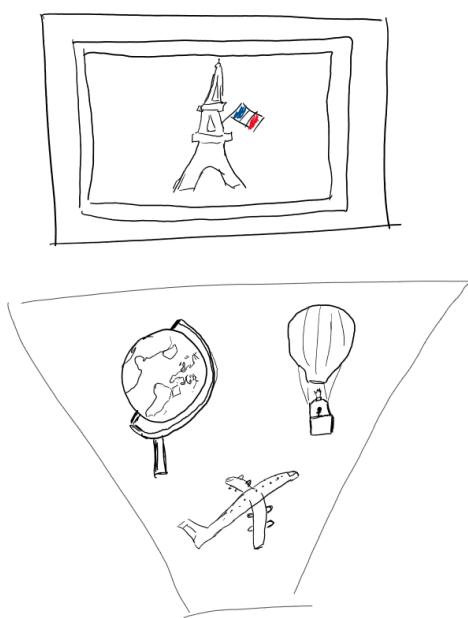


Figura 7.40: Boceto de la decoración para la prueba de turismo. En la pantalla una imagen de la torre Eiffel, con decorado de viajes: avión, globo terráqueo y globo aerostático.

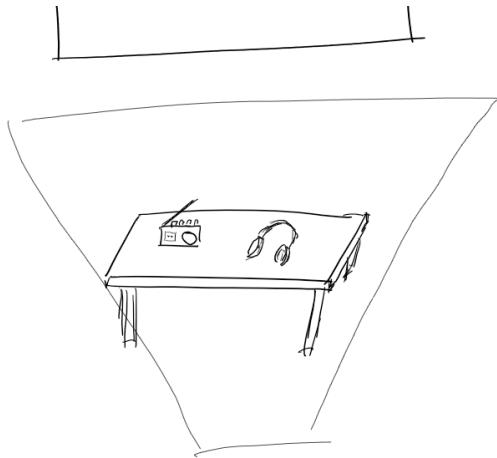


Figura 7.41: Boceto de la decoración para la prueba de adivinar la canción.

gen en la pantalla y el jugador tendrá que discernir de qué objetos se trata. Para facilitar esta tarea es sería posible que en el escenario aparecieran varios objetos virtuales, entre ellos los que se encuentran superpuestos en la imagen de la pantalla.

Pruebas de escenario

Estas son aquellas pruebas que dependen de los objetos del escenario y requieren su interacción para ser completadas, dejando a la pantalla el papel de decoración.

La prueba de figuras, en la que el usuario debe adoptar una posición que se le muestra antes de que se acabe el tiempo, utiliza un objeto que imita a un recortable de cartón o espantapájaros de una silueta posando. Dicha silueta se moverá hacia el usuario y este debe haber adoptado la posición correcta cuando la silueta llegue al jugador. El funcionamiento de esta prueba aparece bocetado en la figura 7.42.

Para la prueba de objetivos, en la que el jugador debe tocar objetos que se mueven hacia él, el escenario utilizado solo contendrá dichos objetos en movimiento, como se pueden ver en el boceto de la figura 7.19.

Para la prueba de asociación de sonidos únicamente se presenta ante el jugador una mesa en la que aparece un grupo de distintos objetos y el jugador debe seleccionar cual está produciendo el sonido mediante botones colocados frente a cada objeto (figura 7.43). De forma similar, la prueba de agrupación de objetos también utiliza solo una mesa con un grupo de objetos, en este caso tiene

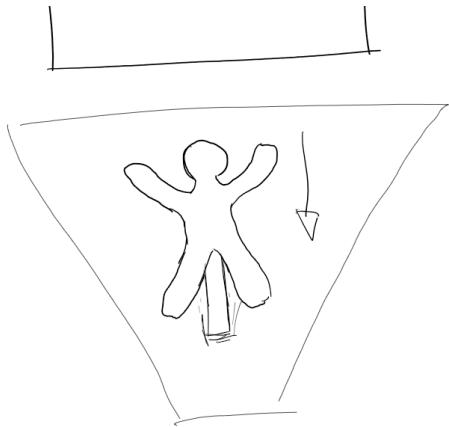


Figura 7.42: Boceto para la prueba de figuras, con una silueta acercándose al jugador.

también dos zonas donde agrupar los objetos por sus categorías (figura 7.44).

Por último, está la prueba de localización de sonidos, en este caso el escenario a utilizar es el propio escenario principal, ya que la prueba consiste en colocar una fuente de sonido en un punto del espacio y que el jugador se gire hasta encontrarla. De este modo se puede aprovechar el amplio espacio del escenario para colocar la fuente como se ve en la figura 7.45.

7.3.3. Iteración 2

En esta iteración se procede a la creación de los distintos escenarios y decorados para el proyecto. Para esto se puede utilizar cualquier software de modelado 3D, incluso el propio Unity. Se ha elegido utilizar Blender, una herramienta totalmente gratuita y de las más potentes del mercado.

Escenario principal

Todas las partes básicas del escenario se han creado en Blender como un único elemento formado internamente por distintas partes. Se ha comenzado creando una plataforma circular donde estará el usuario, a la cual se le ha añadido un suelo inferior también circular. Para completar la parte central de escenario (figura 7.46), se añaden unas gradas en la mitad posterior del círculo. Para crear las gradas se ha recurrido a la técnica de revolución de un perfil, que consiste en diseñar únicamente el perfil del objeto deseado y posteriormente hacerlo girar sobre un eje para crear un objeto en la zona que barre el perfil.

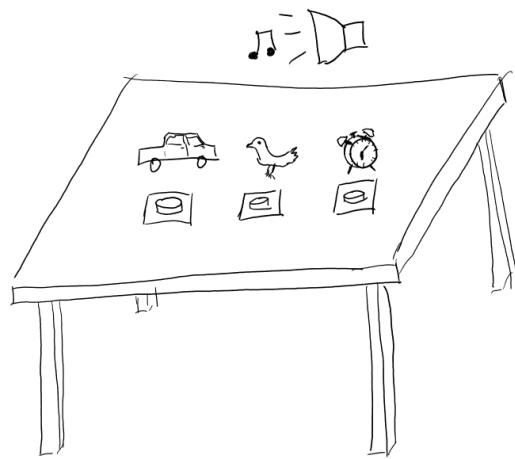


Figura 7.43: Boceto con la mesa para la prueba de asociación de sonidos. Presenta distintos objetos con botones que corresponde a cada uno.

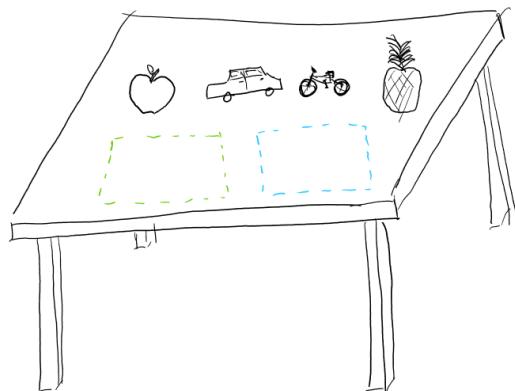


Figura 7.44: Boceto para la prueba de asociación de objetos, mostrando dos zonas (verde y azul) para la clasificación de los objetos.

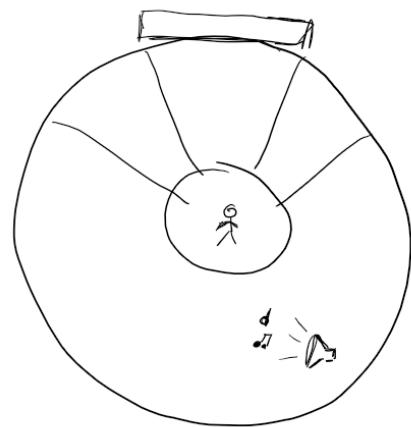


Figura 7.45: Boceto de la vista cenital del escenario principal, mostrando una posible colocación de la fuente de sonido para la prueba de localización de sonidos.

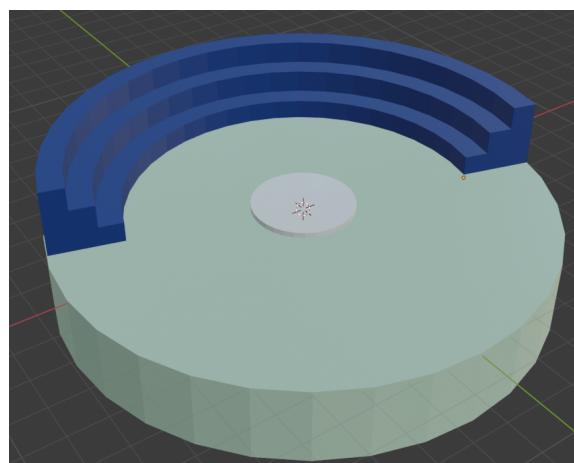


Figura 7.46: Parte central del escenario.

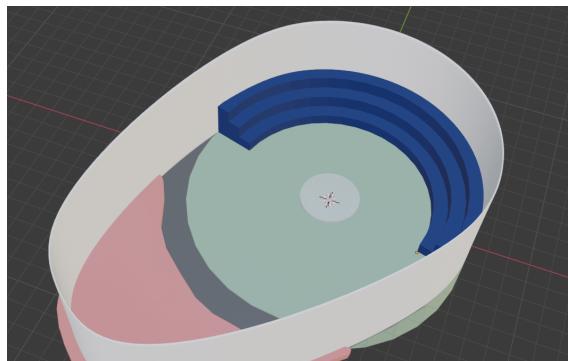


Figura 7.47: Plataforma principal con el escenario de baile y las paredes que rodean todo el perímetro.

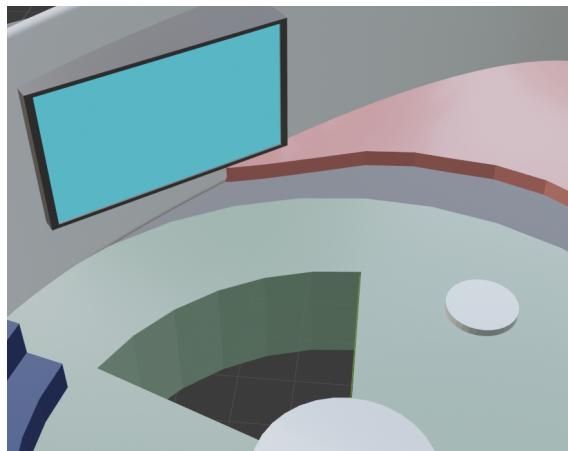
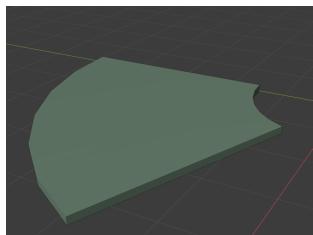
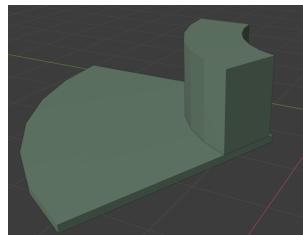
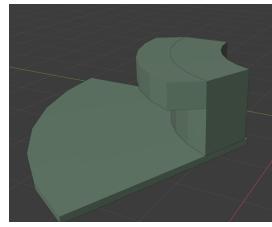
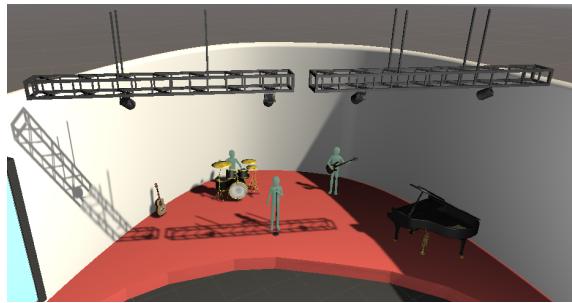


Figura 7.48: Detalle de la colocación de la pantalla, el pedestal secundario y el hueco del ascensor.

Como parte del proceso continuo de mejora del proyecto, se decide que determinadas pruebas transcurran en una zona concreta del escenario con una ambientación más adecuada. Estas pruebas son las que conllevan amplios movimientos del usuario: baile, posiciones, parada y objetivos, por eso, la ambientación de esta nueva zona consistirá en colocar un escenario de baile desde el cual recibirá la prueba el jugador. Se añade también una pared para encapsular el escenario dentro del espacio virtual (figura 7.47).

En el siguiente paso en la creación de la base para el escenario de juego, se añade una pantalla para que el jugador pueda recibir información mediante texto e imágenes, un pequeño pedestal frente al escenario donde el jugador se trasladará para realizar las pruebas pertinentes, y finalmente se ha recortado un agujero en el suelo del escenario por donde bajarán y subirán los decorados de cada prueba individual. Estos cambios se pueden apreciar en la figura 7.48.

Finalmente se construye el ascensor que ocupará el hueco y se usará como

**Figura 7.49:** Ascensor simple.**Figura 7.50:** Ascensor con mesa.**Figura 7.51:** Ascensor con mesa extendida.**Figura 7.52:** Vista del escenario de baile completo.

base para el decorado de las pruebas individuales. Para ello se ha diseñado un ascensor modular que puede adaptarse a las necesidades de cada prueba, pudiendo ser utilizado en su forma básica: proporcionando únicamente un suelo (figura 7.49); llevando integrada una mesa en la que aparecerán objetos con los que el jugador podrá interactuar (figura 7.50); o con una extensión para la mesa que permite colocar mayor cantidad de objetos en ella (figura 7.51).

Escenarios secundarios

El decorado de los escenarios secundarios consta de una mezcla de objetos creados en Blender y una serie de objetos que ofrece Asset Store, la plataforma oficial de Unity para la distribución de contenido para ser utilizado en diversos proyectos.

Para las pruebas de movimiento, hay que decorar el escenario de baile, para esto se han utilizado modelos de distintos instrumentos musicales (guitarra, piano, bajo y trompeta), así como un micrófono. Para dar vida a los músicos se ha utilizado un modelo de humano caricaturizado, que además se utilizará para representar todos los movimientos que el jugador necesita realizar. Por último, se han añadido unas estructuras de iluminación y varios focos en la parte superior del escenario. Véase figura 7.52.

Además, para la prueba de objetivos se han creado dos cubos biselados en

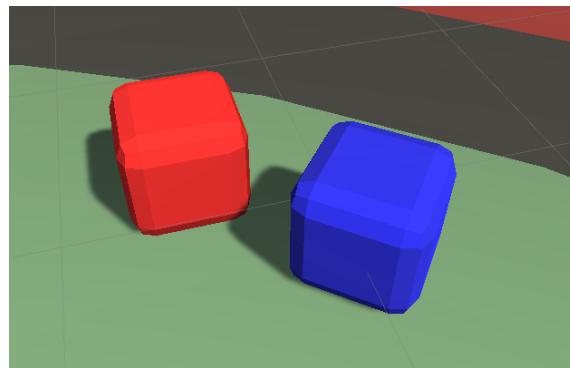


Figura 7.53: Modelos para los objetivos que el jugador debe golpear.

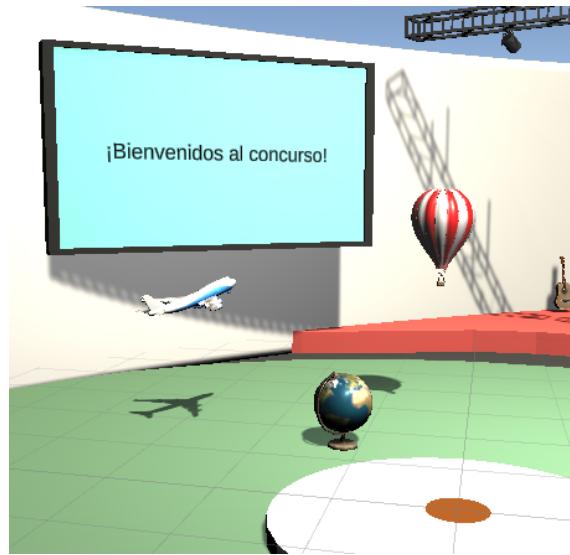


Figura 7.54: Presentación del decorado para la prueba de turismo.

colores azul y rojo (figura 7.53) que representarán los objetivos que el jugador debe golpear.

El decorado para la prueba de turismo utiliza el ascensor sin mesa y adorna el espacio entre el jugador y la pantalla con objetos de temática viajera. En este caso como se ve en la figura 7.54, se han colocado un globo aerostático, un avión de pasajeros despegando y en el centro un globo terráqueo.

Para la prueba donde el jugador debe adivinar una canción que está sonando se ha utilizado el ascensor con mesa, en la cual se han colocado varios objetos del mundo de la música (figura 7.55). En el centro se han colocado unos auriculares de diadema, mientras que a cada uno de sus lados hay un dispositivo cuyo uso es reproducir música: un tocadiscos y una radio. Se han elegido objetos



Figura 7.55: Mesa con decorado para la prueba de adivinar la canción.

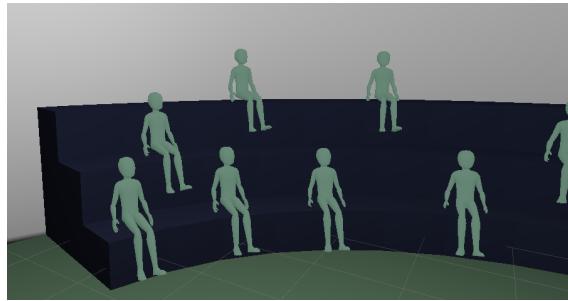


Figura 7.56: Presentación de las figuras que actúan como público.

de estética antigua ya que resultarán más familiares al público objetivo de este proyecto, que son las personas mayores.

El resto de las pruebas utilizan objetos virtuales para su realización, por lo que la inclusión de objetos decorativos similares podría llevar a confusión, por ejemplo, en la prueba de agrupación de objetos. Aun así, se seguirá utilizando el patrón ya marcado de utilizar los tres tipos de ascensor diseñados, pero en lugar de contener decorado, contendrán los objetos necesarios para las pruebas.

Para finalizar la decoración del escenario general y generar un ambiente más acogedor para el jugador, se han poblado las gradas con modelados caricaturizados como los músicos del escenario de baile (figura 7.56).

7.3.4. Iteración 3

En esta última iteración para esta entrega se busca conocer la respuesta de varios usuarios ante los escenarios desarrollados. Parte de esta iteración se ha realizado en paralelo a la anterior, durante el desarrollo de los escenarios. Esto ha permitido realizar un trabajo más efectivo desde una fase temprana de la entrega. Esta vez solo se ha podido contar con una persona para que ofrezca su

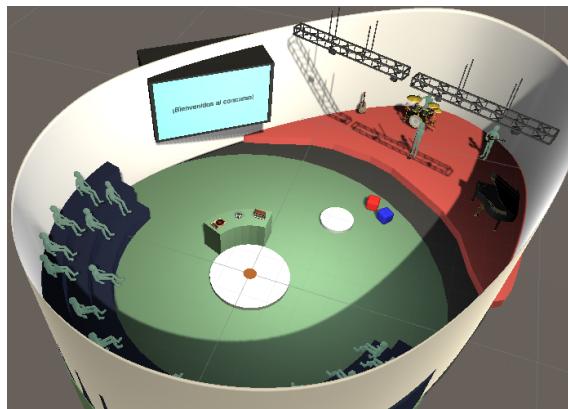


Figura 7.57: Montaje final del escenario para la entrega 3.

opinión y críticas sobre el trabajo desarrollado.

Esta persona tiene entre 20 y 30 años y tiene un amplio conocimiento tecnológico pero una experiencia limitada en realidad virtual. Para la realización de la prueba se ha colocado al usuario en el centro del escenario y se le ha dado libertad de desplazamiento por todo el escenario mientras se iba conversando para obtener información útil.

De esta prueba han surgido varios pequeños cambios que ya han sido realizados en el proyecto, la mayoría relacionados con la posición de ciertos objetos, como la pantalla que estaba demasiado alta. Pero sin duda, el factor más importante para tener en cuenta es la escala y tamaño de los objetos del mundo virtual. Es necesario que la escala del entorno sea lo más parecida a la realidad, ya que de otra forma crea situaciones incómodas como: mesas demasiado altas o bajas, modelos de personas que parecen ser gigantes, o un espacio demasiado grande y vacío.

Finalmente, tras tomar en consideración toda esta información, se ha ido adaptando el escenario hasta su forma final, que se puede ver en la figura 7.57.

7.3.5. Conclusiones de la entrega

Al igual que el desarrollo informático, la creación de arte digital es un proceso complejo y que requiere de conocimientos avanzados para obtener los resultados más realistas posibles. En este caso, para limitar la escala del proyecto ha sido necesario simplificar los diseños y utilizar objetos 3D prediseñados por otras personas, las cuales son acreditadas tanto en esta memoria como dentro del juego.

Se ha intentado recrear de forma esquemática un escenario de concurso televisivo y utilizar objetos con los cuales puedan estar familiarizados los jugadores, en este caso, mayores.

A pesar de que el entorno no es detallista, gracias a la realidad virtual, se consigue aumentar la inmersión. Esto, acompañado de las mejoras sugeridas tras algunas pruebas con personas ajenas, permite que haya una credibilidad suficiente durante el juego, y se determina que los objetos y decorados elegidos son adecuados para las pruebas a realizar y para alcanzar el objetivo de este proyecto.

7.4. Entrega 4

7.4.1. Objetivo

El objetivo de esta entrega es diseñar varias pruebas concretas de cada uno de los tipos y posteriormente añadirlas al proyecto. Una vez estén las pruebas terminadas, será necesario realizar un test con distintos usuarios para ver cómo funcionan y posibles mejoras que se puedan realizar.

7.4.2. Iteración 1

Durante esta iteración se van a diseñar todas las pruebas que habrá en el proyecto. Se ha elegido que haya tres pruebas de cada tipo para mostrar suficiente variedad en ellas, pero que no sean una cantidad demasiado grande para este proyecto.

Pruebas de motricidad

Para cada prueba de baile es necesario elegir una canción y tres pasos. Un paso puede ser una posición concreta de las extremidades, un movimiento concreto entre dos puntos de una extremidad, o también puede ser un momento de libertad en el que el usuario pueda moverse según le plazca al ritmo de la música. En esta prueba se busca estimular el movimiento y flexibilidad del usuario, así como su coordinación. Los momentos de libertad no solo permiten estimular estas capacidades si no que también actúan de vía de expresión para los sentimientos y estado del jugador. Es importante elegir música con la que los mayores estén familiarizados para hacer que se sientan inmersos y cómodos con la can-

ción y el baile. Teniendo todo esto en cuenta se han elegido las siguientes tres canciones junto a sus pasos:

- Ejemplo 1: El chacachá del tren – El consorcio
 - Brazos arriba y abajo alternados
 - Libre
 - Brazos arriba y abajo alternados
- Ejemplo 2: Las flechas del amor – Karina
 - Brazo izquierdo de izquierda a derecha
 - Brazo derecho de izquierda a derecha
 - Libre
- Ejemplo 3: Chica ye-ye – Concha Velasco
 - Brazos arriba y abajo alternados
 - Ambos brazos a la izquierda y derecha a la vez
 - Libre

La siguiente prueba de motricidad es la de figuras, que consiste en que el jugador adopte tres poses concretas de forma encadenada. Para que el jugador tenga tiempo de colocarse en posición, habrá cinco segundos de margen entre cada cambio de posición. Para definir un ejemplo de este tipo de prueba es necesario escoger un conjunto de tres poses corporales. Los tres ejemplos escogidos son:

- Ejemplo 1:
 - Ambos brazos arriba
 - Brazo izquierdo abajo, derecho a la derecha
 - Ambos brazos al frente
- Ejemplo 2:
 - Brazo derecho al frente, izquierdo arriba
 - Brazo derecho en diagonal abajo, izquierdo en diagonal arriba
 - Ambos brazos arriba

- Ejemplo 3:

- Ambos brazos en diagonal abajo
- Ambos brazos al frente
- Ambos brazos en diagonal arriba

Para la última prueba de motricidad, la de parada, el jugador tiene que repetir un movimiento de forma constante y cuando sea necesario parar en seco y mantener la posición. Con esta prueba se estimula aún más la movilidad del jugador además de su tiempo de reacción. En este caso solo es necesario definir el movimiento que el jugador seguirá, ya que el momento en el que tendrá que parar se decidirá aleatoriamente en cada prueba con un intervalo de entre 10 y 15 segundos.

- Ejemplo 1: Mover los brazos de arriba abajo al contrario (brazo derecho arriba mientras el izquierdo está abajo)
- Ejemplo 2: Trazar círculos en el frente con cada una de las manos
- Ejemplo 3: Trazar un círculo con la mano derecha mientras se sube y baja la izquierda.

Pruebas de memoria

En las pruebas de memoria se intenta estimular la memoria a largo plazo del jugador. En la prueba de turismo se estimula la memoria mediante imágenes icónicas de diferentes ciudades y monumentos famosos a nivel mundial. La imagen se muestra en la pantalla principal del escenario del concurso y el jugador tiene que describirla, recordar su nombre, su localización y todo lo que sea capaz de recordar. Para esta prueba se han elegido las siguientes tres imágenes:

- Ejemplo 1: Torre Eiffel, París, Francia. Figura 7.58
- Ejemplo 2: Coliseo, Roma, Italia. Figura 7.59
- Ejemplo 3: Estatua de la libertad, Nueva York, EE. UU. Figura 7.60



Figura 7.58: Torre Eiffel, París, Francia.



Figura 7.59: Coliseo romano, Roma, Italia.



Figura 7.60: Estatua de la libertad, Nueva York, EE. UU.

La prueba de adivinar la canción utiliza el sentido del oído para despertar recuerdos y sentimientos del jugador, por esto es necesario utilizar canciones que sean reconocibles y agradables para las personas mayores cuando estén jugando. En este caso se usarán las siguientes canciones en español:

- Ejemplo 1: Dos gardenias – Antonio Machín
- Ejemplo 2: Que viva España – Manolo Escobar
- Ejemplo 3: La luna enamorada – Los Bocheros

Prueba de lenguaje

En la prueba de situaciones se intenta que el jugador razoné y hable sobre una situación concreta. Para ello se mostrará una imagen en la pantalla del escenario que representará una situación concreta. Se intenta que el jugador esté inmerso en la experiencia y pueda desarrollar sus pensamientos sobre la imagen, por lo que es importante que se traten de situaciones comunes, por las que probablemente hayan pasado y que les evoque sentimientos. Se han elegido estas imágenes:

- Ejemplo 1: Médico vendando la mano de una persona. Figura 7.61
- Ejemplo 2: Madre preparando a su hijo para el colegio. Figura 7.62
- Ejemplo 3: Gente bailando en la Feria de Abril. Figura 7.63



Figura 7.61: Médico vendando una mano.



Figura 7.62: Madre preparando a su hijo para el colegio.



Figura 7.63: Gente bailando en la feria de abril.

Pruebas de razonamiento

En estas pruebas se intenta activar y ejercitar el razonamiento del jugador. La prueba de agrupación de objetos intenta que el jugador descubra las dos categorías ocultas en las que se clasifican los objetos que se le muestran. Es necesario que estos objetos sean fácilmente reconocibles en forma de objeto virtual tridimensional, así como claramente distinguible la categoría a la que pertenecen cuando se presentan con el resto de los objetos. Para cada ejemplo se mostrarán cuatro objetos de dos categorías sobre la mesa del escenario:

- Ejemplo 1:
 - Frutas: Manzana, plátano
 - Ropa: Zapato, guante
- Ejemplo 2:
 - Transporte: Avión, coche
 - Objetos de casa: Bombilla, teléfono
- Ejemplo 3:
 - Plantas: Árbol, flor
 - Música: Guitarra, radio

En la prueba de asociación de sonidos se muestran objetos sobre la mesa de la misma forma que en la prueba anterior, con la única condición de que estos objetos tienen que poder emitir un sonido característico. En este caso, se pueden usar los mismos objetos para los tres ejemplos y lo que cambiará será qué objeto produce el sonido. Para tener variedad de sonidos distinguibles, se han elegido los siguientes:

- Coche
- Pájaro
- Guitarra
- Teléfono

Pruebas de comprensión espacial

En las pruebas de comprensión espacial se intenta evaluar la capacidad del jugador de tener una comprensión firme del espacio que lo rodea así como de los estímulos que recibe de él, principalmente mediante la audición o la visión. La prueba de siluetas presenta al usuario con tres siluetas sombreadas y debe descubrir de qué objetos se trata. Las siluetas irán en conjuntos de tres y podrán o no estar superpuestas. Para no aumentar la complejidad del proyecto, se utilizan modelos ya utilizados en otras pruebas agrupados de la siguiente forma:

- Ejemplo 1: Coche, teléfono, árbol.
- Ejemplo 2: Manzana, bombilla, guante.
- Ejemplo 3: Zapato, plátano, flor.

Finalmente, la prueba de localización de sonidos consiste en colocar una fuente sonora en el espacio al rededor del usuario. El sonido será una persona hablando, ya que es un estímulo muy habitual en la vida real y las posiciones donde aparecerá se elegirán de forma aleatoria en cualquier punto del escenario fuera del campo visual del jugador.

7.4.3. Iteración 2

El objetivo de esta iteración es obtener un prototipo del juego que contenga varias pruebas de cada tipo. Se centra la atención en que las pruebas sean completamente funcionales y suficientemente variadas entre ellas.

Las pruebas implementadas en el proyecto tienen una estructura básica que funciona de igual forma para todas: da comienzo la prueba, se baja el ascensor actual, se sube el correspondiente a la prueba, se instancian todos los objetos necesarios para la prueba, se comprueba su finalización y finalmente se revierte al ascensor principal y se destruyen los objetos que no sean necesarios. Cada una de las pruebas solo se diferencia del resto en elementos concretos como:

qué objetos y dónde se instancian, si tiene que reproducirse algún sonido o la mecánica concreta de la prueba.

En cuestión de código, esto se traduce en una estructura de clases con herencia, donde el funcionamiento concreto de cada prueba viene dado por su propia clase que a su vez hereda toda la funcionalidad general de una clase superior llamada en este caso 'Prueba'. La clase 'Prueba' es la que contiene la mayoría de la funcionalidad, como se ve en la figura 7.64, dejando a las clases individuales únicamente como medios se preparar todo lo necesario para dicha prueba y posteriormente dar comienzo a la misma. En la figura 7.65 se puede ver un ejemplo de como se indican los objetos que se tienen que instanciar (añadiéndolos a una lista), el ascensor correspondiente que debe aparecer y el sonido que se reproducirá.

Utilizando esta estructura de clases se potencian de forma simultánea dos aspectos del proyecto:

- Manteniendo la funcionalidad principal agrupada en una clase, se facilita la inclusión de nuevas pruebas en el futuro, siendo solo necesario crear la clase hija que prepare la prueba para su correcta ejecución.
- Tener una estructura de herencia implica que en cualquier momento una clase hija puede sobrescribir el funcionamiento de la clase original, permitiendo la implementación de pruebas completamente distintas y más complejas que las actuales.

Pruebas de motricidad

Para las pruebas que ejercitan la motricidad se ha utilizado la funcionalidad de los 'Triggers' creados durante el apartado 7.2.3.

Cuando la prueba de baile da comienzo, la clase correspondiente carga la estructura de los triggers (posiciones que debe adoptar el jugador) desde un fichero JSON. También comienza a reproducir la canción requerida para la prueba correspondiente. La prueba de posiciones utiliza la misma técnica pero cargando otras estructuras diferentes y sin música, mientras que la prueba de paradas hace lo mismo pero en lugar de reproducir una canción, se prepara para reproducir un sonido que indica al jugador que tiene que detenerse.

En estas pruebas es importante la forma en la que se le transmite al usuario que posición debe adoptar. El jugador tiene que tener claro dónde tiene que colocar sus extremidades tanto vertical como horizontalmente. Un símbolo como

```
public class Prueba : MonoBehaviour
{
    protected GameObject ascensor;
    protected GameObject imagen;
    protected GameObject sonido;
    protected string pathTriggers;
    protected int botonCorrecto;
    protected List<GameObject> listaObjetos;

    private scriptAscensor ascensorPrincipal;
    private scriptAscensor nuevoAscensor;

    private bool activo;
    private bool terminar;
    private bool listo;
    private bool objetosCreados;

    ④ Mensaje de Unity | 0 referencias
    void Start()...

    ④ Mensaje de Unity | 0 referencias
    void Update()...

    14 referencias
    public virtual void CargarPrueba()...

    7 referencias
    public void MoverAscensores()...

    10 referencias
    public void TerminarPrueba()...

    1 referencia
    public void CargarImagenPantalla(GameObject spr)...

    1 referencia
    public void CargarSonido(GameObject sonido)...

    1 referencia
    public void CargarObjetos(List<GameObject> objetos)...

    1 referencia
    public void PruebaCorrecta()...
}
```

Figura 7.64: Estructura de la clase 'Prueba'.

```
14 referencias
public class pruebaSonidos : Prueba
{
    public override void CargarPrueba()
    {
        ascensor = (GameObject)Resources.Load("Escenarios/Esc_sonidos");
        listaObjetos = new List<GameObject>();

        listaObjetos.Add((GameObject)Resources.Load("Objetos/Coche"));
        listaObjetos.Add((GameObject)Resources.Load("Objetos/Pajaro"));
        listaObjetos.Add((GameObject)Resources.Load("Objetos/Guitarra_no"));
        listaObjetos.Add((GameObject)Resources.Load("Objetos/Telefono"));

        switch (Random.Range(0, 3))
        {
            case 0:
                sonido = (GameObject)Resources.Load("Canciones/son_pajaro");
                botonCorrecto = 4;
                break;
            case 1:
                sonido = (GameObject)Resources.Load("Canciones/son_guitarra");
                botonCorrecto = 2;
                break;
            case 2:
                sonido = (GameObject)Resources.Load("Canciones/son_telefono");
                botonCorrecto = 1;
                break;
            default:
                break;
        }

        MoverAscensores();
    }
}
```

Figura 7.65: Sobrescritura del método 'CargarPrueba' para la prueba de asociación de un sonido con un objeto.

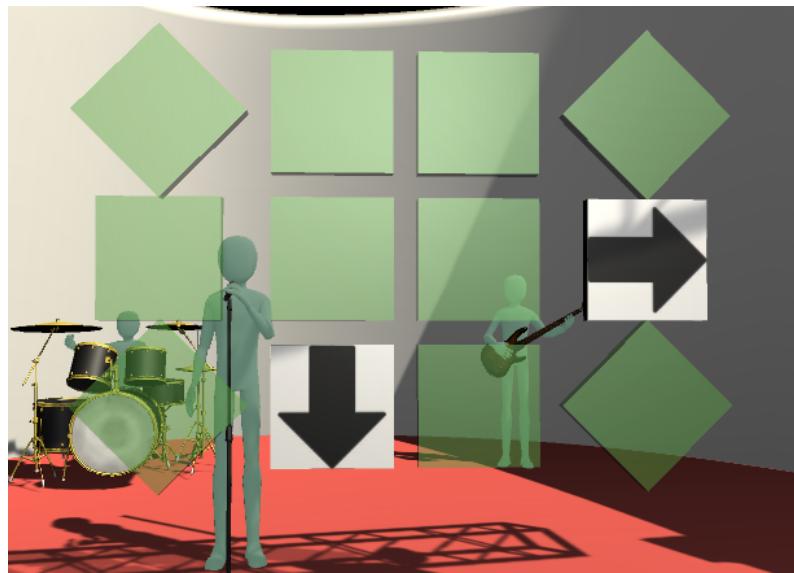


Figura 7.66: Matriz de señales que indican al usuario dónde colocar sus manos.

una flecha es un ícono fácilmente reconocible y con un significado muy claro si se utiliza correctamente. Teniendo en cuenta esto, un método para transmitir la información al jugador podría ser mediante flechas que aparecen delante del mismo y que indican dónde colocar las manos.

Con la idea de reflejar lo mejor posible las indicaciones al usuario, se puede utilizar una estructura que se asemeje a la propia estructura de triggers utilizada para detectar los movimientos. Mediante este paralelismo se puede mejorar la precisión y facilidad con la que el usuario capta y entiende los movimientos que tiene que hacer. De este modo, se ha creado una matriz de cuadrados con flechas que aparece frente al usuario y cambian de la misma forma que cambian los triggers (véase figura 7.66).

Pruebas de memoria

Las pruebas de memoria se basan en presentar una imagen o canción al jugador para hacerle recordar información sobre ellas o revivir experiencias personales que tenga asociadas con ellas. Para la reproducción de las canciones se utiliza un objeto nativo de Unity llamado Audio Source. Es un objeto que permite reproducir un clip de audio dentro de la escena del juego. Por lo tanto, solo es necesario indicar qué clip de audio sonará durante cada prueba como se muestra en la figura 7.31. Esta misma técnica es la que se utiliza para todas las pruebas que necesiten reproducir algún sonido.

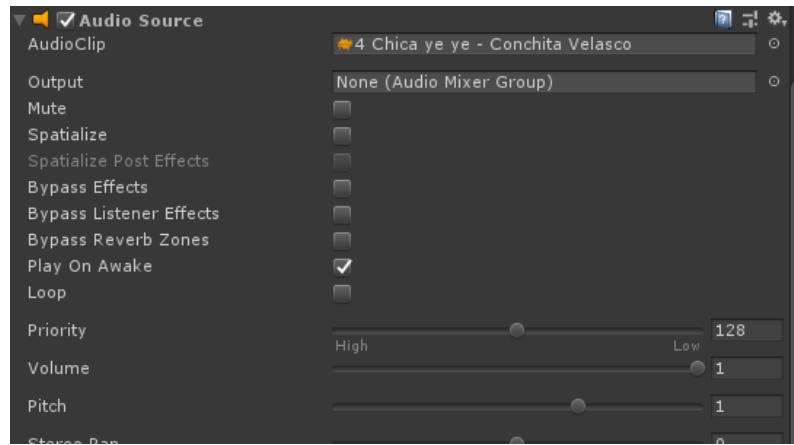


Figura 7.67: Audio Source de Unity mostrando el clip actual que reproduce, así como otros de sus parámetros.

Para las pruebas en las que se presenta al jugador una imagen, se utiliza una primitiva de Unity que permite crear planos bidimensionales dentro de una escena 3D. Utilizando la pantalla gigante que forma parte del escenario se coloca sobre ella un plano en el que posteriormente se puede colocar como textura del mismo cualquier imagen. De esta forma, solo es necesario activar y desactivar el plano junto con su textura correspondiente. En la imagen 7.68 se representa un objeto de tipo plano, que muestra una imagen del coliseo romano.

Prueba de lenguaje

En esta prueba se presenta al jugador una imagen en la que aparece una situación cotidiana, que puede haber vivido, con el objetivo de que hable y haga comentarios sobre ella. Para esto solo es necesario mostrar una imagen en la pantalla gigante del escenario, siguiendo la misma técnica usada en la prueba anterior.

Pruebas de razonamiento

Para las pruebas de razonamiento se utiliza una mesa frente al jugador en la que aparecen varios objetos con los que debe interactuar para superar el reto. En la prueba de agrupación de objetos (véase figura 7.69), además es necesario crear dos zonas donde categorizar los objetos. Estas zonas pertenecen a la propia mesa y siempre son las mismas. Para cada ejemplo de esta prueba solo es necesario cambiar los objetos que se instancian. Para asegurar que todos los objetos aparecen en el lugar adecuado, se han creado cuatro objetos vacíos, asociados a la mesa, que sirven únicamente como punto de referencia. A la hora



Figura 7.68: Parámetros de un objeto plano de Unity mostrando una imagen del coliseo romano.

de instanciar los objetos, se buscan estas referencias y se genera un objeto en la posición de cada una.

En la figura 7.70 se puede ver la estructura de objeto para la mesa, que está compuesta por dos piezas (suelo y mesa), así como dos conjuntos de objetos, el primero de ellos, las 'SpawnZone' que marcan los puntos donde aparecerán los objetos; y el segundo son las plataformas donde deben colocarse los objetos para considerarse clasificados.

En la prueba de asociación de sonidos, en lugar de necesitar zonas para clasificar los objetos es necesario crear cuatro botones, uno correspondiente a cada objeto (véase figura 7.71). Cada uno de estos botones puede ejecutar un método del código cuando se pulsa. Esto es lo que se utiliza para detectar la respuesta correcta, ya que en tiempo de ejecución, cuando se crean los objetos, automáticamente se asigna que el botón adecuado llame al método para dar la prueba por correcta, como se puede ver en la figura 7.72.

Pruebas de comprensión espacial

Para la prueba de localización de sonidos se usa la misma técnica para la reproducción de audio usada en el resto de pruebas, pero con ligeras diferencias. En el objeto Audio Source que se encarga de reproducir el sonido, es nece-

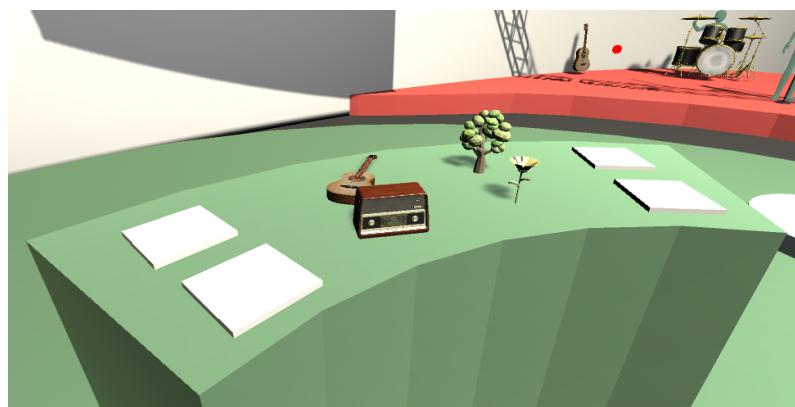


Figura 7.69: Ejemplo de prueba de agrupación de objetos.

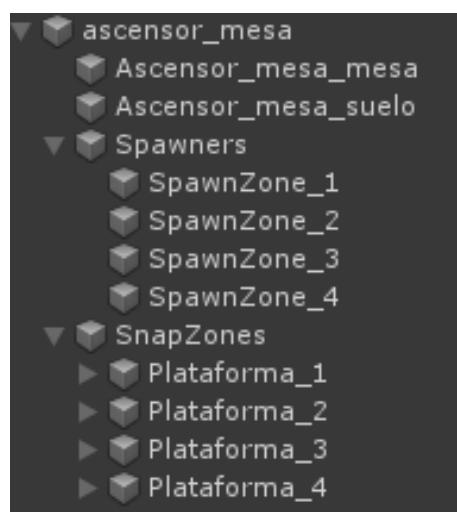


Figura 7.70: Estructura de objeto para la mesa.

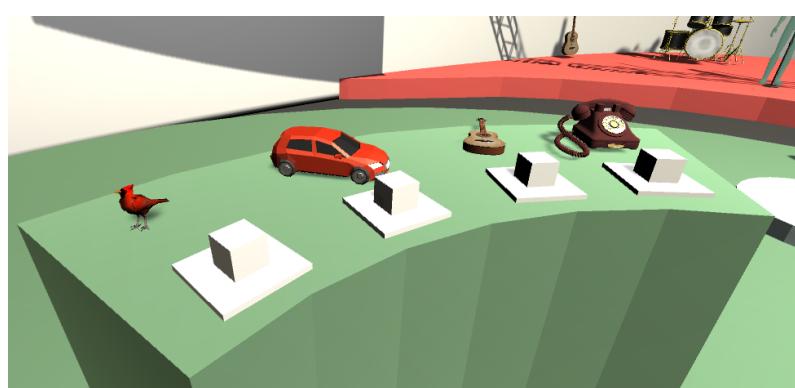


Figura 7.71: Ejemplo de prueba de asociación de sonidos.

```
if (botonCorrecto > 0)
{
    GameObject boton = GameObject.Find("Button_" + botonCorrecto);
    boton.GetComponent<HoverButton>().onButtonIsPressed.AddListener(delegate { PruebaCorrecta(); });
}
```

Figura 7.72: Código que asigna la llamada al método para terminar la prueba de forma correcta al botón adecuado.

rio activar el parámetro 'Spatialize' (ya visto en la figura 7.67), ya que es el que permite oír el sonido en 3D. La otra diferencia es que la fuente de sonido puede aparecer en cualquier punto del espacio al rededor del jugador salvo a su frente. Para esto último se generan en tiempo de ejecución tres valores en ciertos intervalos controlados para acotar el espacio posible.

7.4.4. Iteración 3

En esta última iteración se busca que un usuario real realice las distintas pruebas para poder evaluarlas, encontrar posibles problemas y solucionarlos. Para ello se cuenta con una persona joven, con conocimientos tecnológicos básicos. Esta persona ha repetido varias veces cada una de las pruebas creadas en este proyecto y de su experiencia se pueden sacar las siguientes conclusiones sobre las pruebas:

- Durante las pruebas de baile y figuras, en las que el jugador debe colocar sus manos en posiciones determinadas, es necesario reevaluar el tiempo entre cada pose, ya que a veces este tiempo es demasiado largo y en otras ocasiones, dependiendo de la pose, resulta un poco corto.
- En la prueba de asociación de objetos es necesario ajustar la posición de cada objeto y zona para que el jugador llegue cómodamente a todas ellas. Además, actualmente, no hay forma de recuperar un objeto en caso de que cayera a un lugar inaccesible.
- La elección de canciones para las pruebas de baile y adivinar la canción, parece ser muy buena.
- Los sonidos usados en la prueba de asociación de sonidos son claros y fácilmente distinguibles, lo cual es bueno.
- La posición aleatoria de la fuente de sonido en la prueba donde hay que localizarla puede ser problemática. Hay zonas en las que es muy difícil distinguir la procedencia del sonido.

7.4.5. Conclusiones de la entrega

A pesar de tener las mecánicas básicas implementadas en la entrega 2, esta entrega ha sido la más compleja hasta ahora y ha requerido el mayor tiempo. Especialmente, crear la estructura de clases y toda la funcionalidad básica que trabaje en conjunto y adecuadamente para los distintos tipos de pruebas.

Ha sido particularmente importante poder encontrar canciones e imágenes que el público objetivo del juego (personas mayores) pueda reconocer sin problema y que además, las canciones sean capaces de animarlos o despertar algún sentimiento o memoria.

Finalmente se han encontrado algunos problemas, como que es necesario ajustar el tiempo entre cada posición en las pruebas de motricidad. Posiblemente se necesite más tiempo del que se deja en principio, pero lo ideal sería poder variar este tiempo de forma dinámica para adaptarse a cada jugador. La prueba de localización de sonidos también tiene un problema dependiendo de la posición en la que aparezca la fuente de sonido. Por esto, es mejor cambiar la forma en la que se coloca dicha fuente: en lugar de utilizar una posición aleatoria se pueden definir una serie de posiciones que se consideren adecuadas, de las cuales se elegirá entre una de ellas al comenzar la prueba, asegurando así que la localización será siempre adecuada.

7.5. Entrega 5

Cuestionarios y pruebas. Conclusiones

7.5.1. Objetivo

Esta última entrega del proyecto tiene como objetivo unir todo el trabajo realizado en las anteriores para formar un prototipo de juego completo y jugable, para posteriormente realizar una serie de pruebas con gente real para evaluar la funcionalidad y usabilidad de este proyecto.

7.5.2. Iteración 1

El objetivo inicial de esta entrega era el de diseñar y crear todo el flujo que seguirá el juego. Teniendo en cuenta de que se pretende recrear un concurso televisivo, se va desarrollar una introducción que describa el concurso y presente

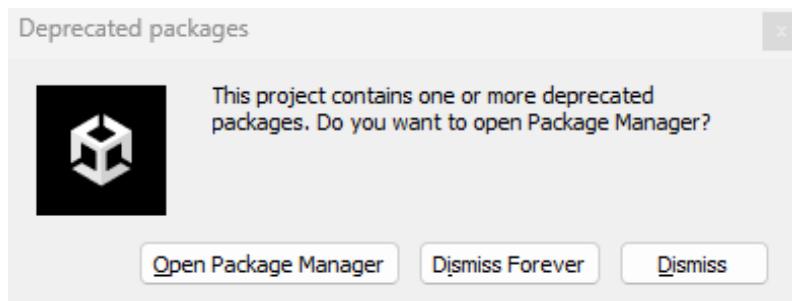


Figura 7.73: Aviso sobre paquetes obsoletos.

las reglas al jugador, además de las fases que ocurrirán al principio y final de cada prueba, así como entre cada una de ellas. Finalmente, se añadirá un menú en el que se podrán cambiar los ajustes del juego, entre ellos: la dificultad, los tipos de pruebas que aparecerán, si el jugador está sentado o de pie, o si el jugador tendrá ayuda de una persona externa o no.

Este objetivo se ha visto alterado por pasar a disponer de un nuevo dispositivo de RV para el desarrollo, en este caso las gafas Meta Quest 2. Como se explica en la sección **Tecnología a usar** (4.1), este dispositivo es mucho más deseable para este proyecto, ya que permite prescindir de un ordenador externo al que conectar las gafas, del uso de estaciones base separadas para realizar el seguimiento del casco y los mandos, y al ser más reciente permite utilizar las bibliotecas más modernas para facilitar el desarrollo. Por estas razones, aún suponiendo un extra de carga de trabajo, se decide realizar el cambio de dispositivo, el cual se describe a continuación.

Migración dispositivo VR

El primer paso es crear una copia de seguridad en caso de haber alguna incompatibilidad. A continuación, se descarga la última versión del editor de Unity, en este caso se usará la versión 2023.1.9f1. Una vez instalada, abrimos el proyecto usando la nueva versión. Durante la migración de una versión a otra aparecen errores y se recibe el aviso de que el proyecto puede no funcionar. Cuando se abre el proyecto aparece un nuevo aviso sobre varios paquetes que están obsoletos (7.73)

Tras desinstalar los paquetes obsoletos, en este caso tanto SteamVR, Zinnia y VRTK, se va a proceder a instalar las nuevas bibliotecas para Meta Quest 2, para ello se va a seguir la documentación oficial. [77] El proceso a seguir es sencillo y consiste en importar el nuevo paquete OculusIntegration (versión 55 en este caso). No ha surgido ningún nuevo problema, la documentación es buena y muy

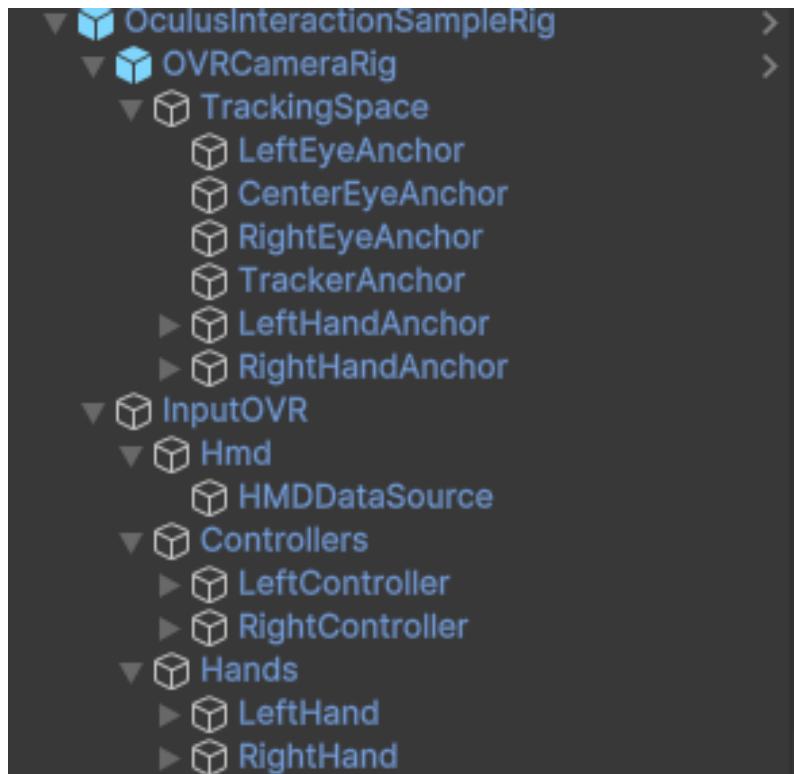


Figura 7.74: Jerarquía del objeto OculusInteractionSampleRig incluido con el SDK de Meta.

visual, y un proceso similar ya se ha descrito en la **Entrega 1** (7.1.2), por lo que no voy a desarrollar mucho esta instalación.

Una vez instalada la nueva biblioteca, el siguiente paso es recuperar todas las funcionalidades que ya estaban desarrolladas para las HTC Vive anteriormente. Por suerte el SDK de integración de Oculus es muy completo y tiene ya implementado casi todo lo necesario. Usando el objeto OculusInteractionSampleRig (figura 7.74), podemos añadir a la escena tanto la cámara como los controladores y todo lo necesario para el seguimiento y la interacción básica con el juego.

Para facilitar el uso del juego a personas con pocos conocimientos tecnológicos, se va a usar el seguimiento de manos proporcionado por el visor, de esta forma se proyecta en el juego una representación exacta de las manos de los usuarios y sus movimientos, como se puede ver en la figura 7.75.

A continuación es necesario añadir las posibles interacciones de las manos con el entorno, en este caso: coger y pulsar objetos y poder seleccionar elementos lejanos mediante un rayo. De nuevo, el SDK proporciona elementos ya configurados para estos casos. Solo es necesario añadirlos a la jerarquía según la figura 7.76. En este caso se añaden tanto para las manos como para los

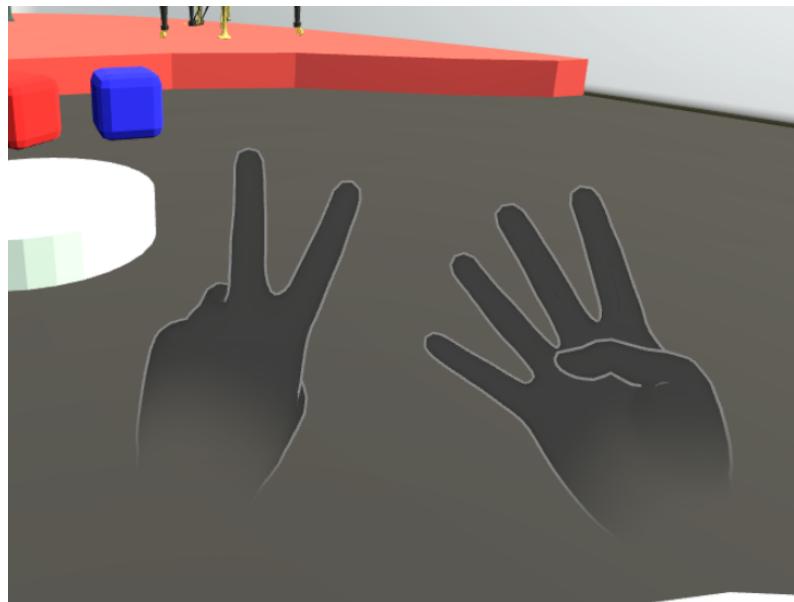


Figura 7.75: Manos virtuales captadas usando el seguimiento de Meta Quest 2.

mandos, en caso de desear utilizarlos.

Finalmente, para que un objeto del juego pueda interactuar con el jugador, necesita nuevos scripts también proporcionados por el SDK, en este caso son:

- *Grabbable*: Permite que un objeto pueda ser cogido y define su comportamiento.
- *Grab Interactable*: Permite que el elemento sea cogido con los mandos.
- *Hand Grab Interactable*: Permite que el elemento sea cogido con las manos.
- *Ray Interactable*: Permite lanzar un rayo sobre el objeto para interactuar con él.

Tras estos pasos, la migración queda terminada y podemos comenzar a utilizar Meta Quest 2 y su seguimiento de manos en el proyecto. Aunque aún es necesario cambiar el funcionamiento de varias pruebas debido a cambios en las nuevas bibliotecas.

En concreto es necesario actualizar las zonas de *snap* utilizadas en las pruebas de asociación, donde el jugador debe coger y colocar en plataformas distintos objetos, agrupándolos por categorías. Lo que se ha decidido hacer es reemplazar este sistema por otro más sencillo en el que un *trigger* sobre la plataforma

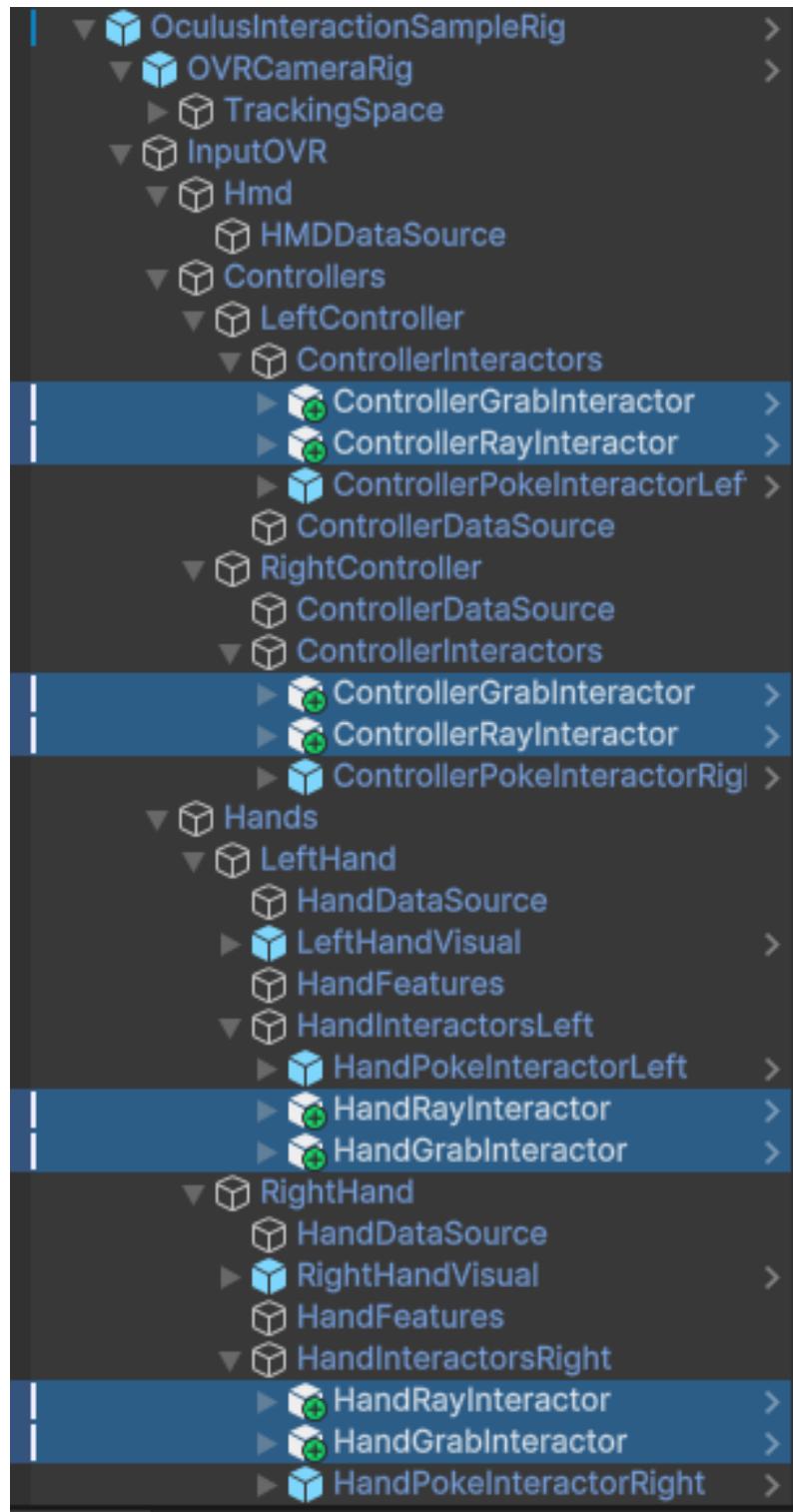


Figura 7.76: *Interactors* que permiten coger, pulsar o lanzar rayos.

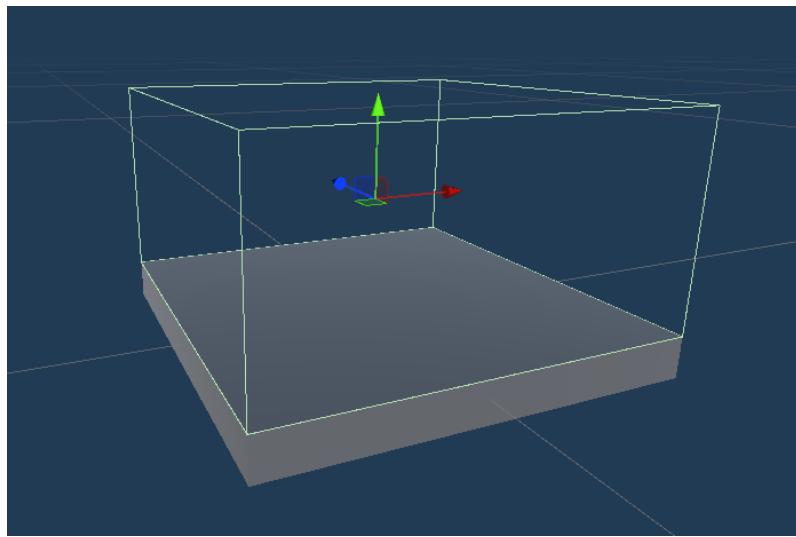


Figura 7.77: Zona de detección formada por un *trigger* sobre una plataforma.

detecta los objetos que se introducen en él (figura 7.77), comprobando si tienen o no la categoría deseada.

Puesto que se usa el mismo método de detección para las dos posibles categorías, es necesario tener en cuenta de qué categoría es el objeto y si las dos zonas de detección contienen el mismo tipo de objeto. Esto se hace por medio de los métodos *OnTriggerEnter* y *OnTriggerExit*. También es necesario llevar la cuenta de cuantos objetos hay dentro del *trigger*. Todo esto se hace como se muestra en la figura 7.78. El funcionamiento es muy similar al método anterior pero simplificado y eliminando el uso de la anterior biblioteca.

De esta manera termina la migración desde HTC Vive y VRTK a Meta Quest 2 y su SDK de interacción. Este cambio se ha hecho para favorecer la facilidad de uso del juego gracias a eliminar la necesidad de cables y al uso de la tecnología de seguimiento de manos de Meta, así como la modernización y actualización de software y dispositivos con mayor soporte y vida útil en el futuro.

Uniendo y completando el proyecto

En este apartado se va a detallar el trabajo realizado para unir todas las pruebas desarrolladas anteriormente creando un flujo de juego completo y repetible. En las anteriores entregas se han ido desarrollando funcionalidades según las necesidades puntuales para alcanzar el objetivo de cada una de ellas, pero llegado este punto es necesario adquirir una visión completa del proyecto y plantear un buen diseño de clases que permitan una buena funcionalidad sin crear pro-

```
④ Mensaje de Unity | 0 referencias
private void OnTriggerEnter(Collider other)
{
    detectedObjects.Add(other.gameObject);
    if (other.gameObject.tag == categoria)
    {
        _correcto = true;
        _numObjetosCategoria++;
    }
    else if (other.gameObject.tag == categoriaSecundaria)
    {
        _correctoSecundaria = true;
        _numObjetosCategoriaSecundaria++;
    }
}

④ Mensaje de Unity | 0 referencias
private void OnTriggerExit(Collider other)
{
    detectedObjects.Remove(other.gameObject);
    if (other.gameObject.tag == categoria)
    {
        _numObjetosCategoria--;
        if (_numObjetosCategoria < 1)
        {
            _correcto = false;
        }
    }
    else if (other.gameObject.tag == categoriaSecundaria)
    {
        _numObjetosCategoriaSecundaria--;
        if (_numObjetosCategoriaSecundaria < 1)
        {
            _correctoSecundaria = false;
        }
    }
}
```

Figura 7.78: Código de la clase DetectZone que gestiona cada zona de detección.

blemas o incompatibilidades en el futuro.

Se ha creado un diseño basado en dividir responsabilidades y asignar cada una su respectiva clase manager, por ejemplo, la clase UIManager se encarga de todo lo relacionado con la interfaz de usuario (UI, User Interface en inglés). A su vez, todas estas clases manager se comunican con la clase GameManager, que es la que controla todo el flujo y realiza las llamadas al resto de managers. También existe la clase Prueba, creada para agrupar las funciones comunes y de la que heredan las clases concretas para cada prueba. A parte de estas clases, también existen clases más específicas para controlar aspectos concreto como el movimiento de los ascensores o la interacción de los objetos en las zonas de detección.

Game Manager Esta es la clase principal que controla el flujo del juego. Cuando la aplicación arranca, busca y guarda las instancias del resto de managers y da la señal al UIManager de que muestre los mensajes de tutorial y bienvenida. Más adelante se encargará también de hacer que el PruebasManager comience la prueba en el momento adecuado, llevar la cuenta del numero de pruebas realizadas o coordinar la interfaz de usuario con el gestor de las pruebas.

Contiene métodos para comenzar y terminar las pruebas (llamando también a los otros managers), saltar pruebas, elegir las posibilidades que se le mostrarán al usuario (véase figura 7.79) y también se encarga de controlar en todo momento el estado en el que se encuentra el juego. De esta forma puede llevar a cabo diferentes acciones dependiendo del estado actual. Estos estados cambian según el grafo de la figura 7.80, se definen en un enumerable llamado EstadoJuego, y son los siguientes:

- **Arrancado.** Mientras el juego está preparando todo lo necesario después de iniciararlo.
- **Tutorial.** El juego está mostrando su tutorial al jugador.
- **Listo.** Estado base del juego, se muestra la pantalla principal en el escenario.
- **EleccionPrueba.** El jugador está eligiendo qué prueba realizar.
- **Prueba.** La prueba está en curso.
- **FinPrueba.** El jugador ha superado la prueba y puede continuar el juego.
- **Final.** Tras superar todas las pruebas, el juego ha terminado. El jugador puede reiniciar si así lo desea.

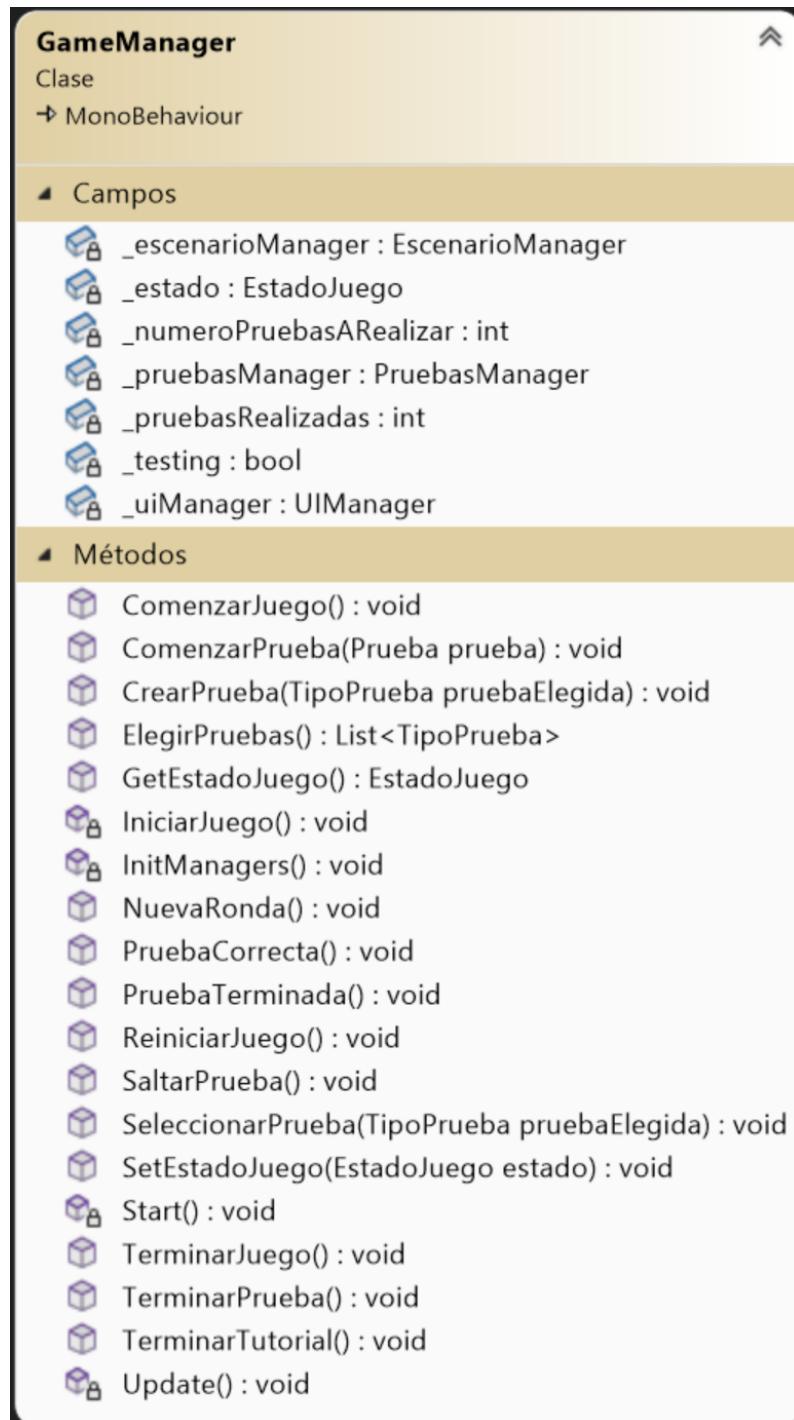


Figura 7.79: Diagrama de la clase GameManager.

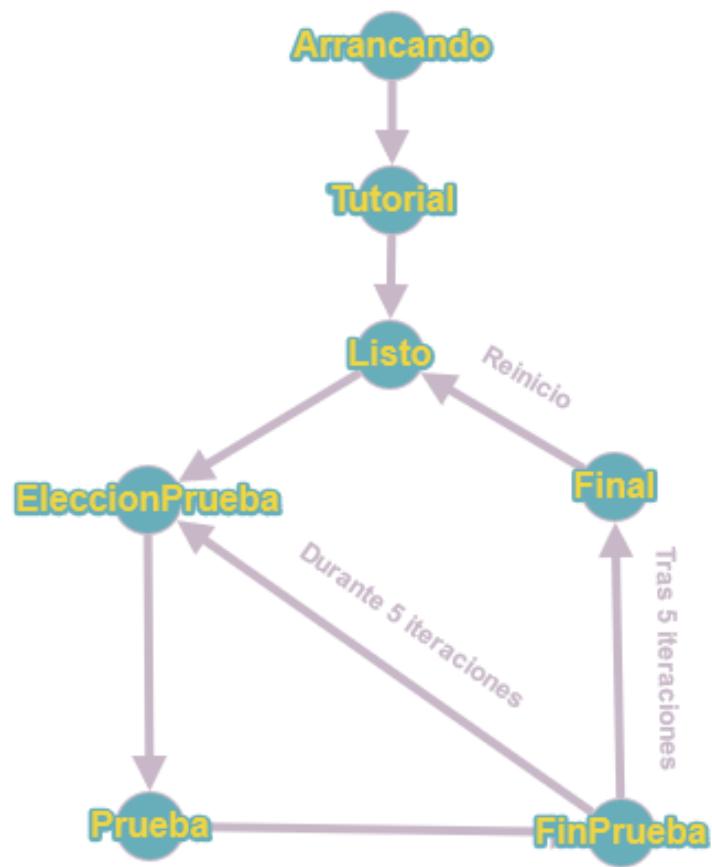


Figura 7.80: Grafo dirigido de los posibles cambios de estado durante el juego.

UI Manager Esta clase se encarga de gestionar todo lo relacionado con la interfaz de usuario. En este caso, la UI se muestra siempre en la pantalla gigante del escenario y el usuario puede interactuar con los botones de ella mediante rayos. Cuando comienza el juego, muestra el tutorial y se mantiene a la espera de que el usuario pulse el botón de continuar, en ese momento, actualiza la pantalla para mostrar la pantalla principal. En la fase de elección de pruebas, muestra los botones para que el usuario elija (figura 7.81), y durante la fase de prueba puede mostrar un mensaje personalizado para cada una. Esta clase controla todos los botones que aparecen en el juego, y al pulsarlos realiza las acciones necesarias, como llamar al GameManager para avanzar de fase, o al PruebasManager para comprobar si una respuesta es correcta.

En el diagrama de clase de la figura 7.82 se muestra como esta clase contiene una referencia a todos los elementos de la interfaz de usuario y los métodos para cambiar la pantalla dependiendo de la fase de juego actual, como PantallaElección() o PantallaFinPrueba().

Pruebas Manager Este controlador se encarga de gestionar todo lo referente a las pruebas que se realizan en el juego. Contiene una referencia a una instancia de cada tipo de prueba (atributos con prefijo 'Prefab' en la figura 7.83), y se comunica con el GameManager para crear, comenzar y terminar las pruebas, así como con el EscenarioManager que dependiendo del tipo de prueba deberá adecuar el escenario como se verá a continuación. En todo momento, este manager contiene una referencia a la prueba que se está realizando y puede comprobar periódicamente si la prueba ha sido completada correctamente con el método CheckPruebaCorrecta().

Escenario Manager Esta clase se encarga de realizar los cambios de escenario durante el juego. Estos cambios son principalmente bajar el ascensor que haya en el momento, crear uno nuevo adecuado a la prueba que se vaya a realizar, incluyendo todos los objetos necesarios para ella, y elevarlo de nuevo a la posición superior frente al jugador. Además, como se puede ver en el diagrama de la figura 7.84, también se encarga de instanciar, mostrar y destruir las imágenes y sonidos utilizados en algunas pruebas, y especialmente se comunica con el TriggerManager y el DetectManager ya que estas funciones de las pruebas forman parte del escenario.

Trigger Manager Esta clase se utiliza en las pruebas de baile y de posiciones. En estas pruebas, el jugador debe seguir unas indicaciones y colocar los brazos

```
0 referencias
public void btnEntendido()
{
    _gameManager.TerminarTutorial();
}

2 referencias
public void MostrarBotonesPruebas(List<TipoPrueba> pruebasAMostrar)
{

    PantallaEleccion();
    for (int i = 0; i < pruebasAMostrar.Count; i++)
    {
        switch (pruebasAMostrar[i])
        {
            case TipoPrueba.Turismo:
                _btnTurismo.SetActive(true);
                break;
            case TipoPrueba.Cancion:
                _btnCancion.SetActive(true);
                break;
            case TipoPrueba.Asocciacion:
                _btnAsociacion.SetActive(true);
                break;
            case TipoPrueba.Posiciones:
                _btnPosiciones.SetActive(true);
                break;
            case TipoPrueba.Situaciones:
                _btnSituaciones.SetActive(true);
                break;
            case TipoPrueba.Baile:
                _btnBaile.SetActive(true);
                break;
            case TipoPrueba.Sonidos:
                _btnSonidos.SetActive(true);
                break;
            case TipoPrueba.LocalizacionSonidos:
                _btnLocalizacionSonidos.SetActive(true);
                break;
            default:
                break;
        }
    }
}
```

Figura 7.81: Código de la clase UIManager. Método para terminar el tutorial pulsando el botón de 'Entendido' y método para cargar los botones de elección de pruebas.

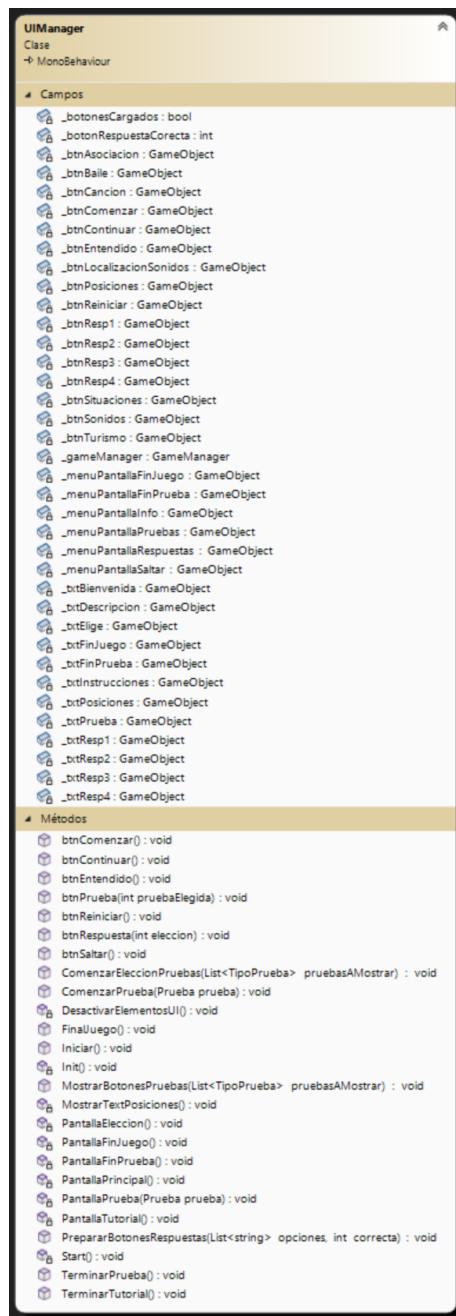


Figura 7.82: Diagrama de clase para UIManager.

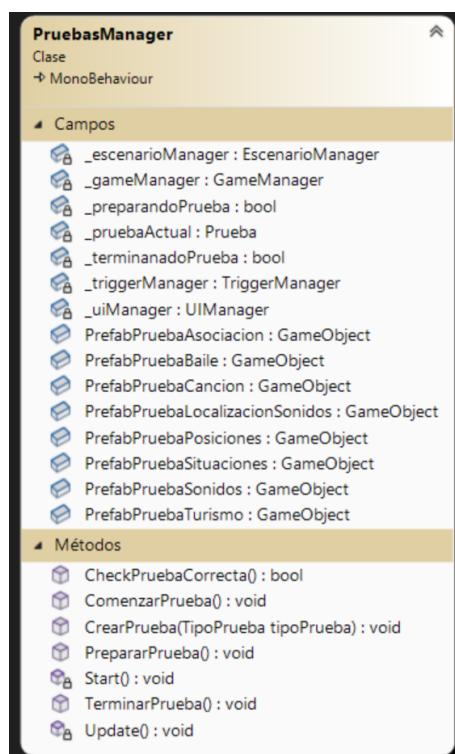


Figura 7.83: Diagrama de clase para PruebasManager.

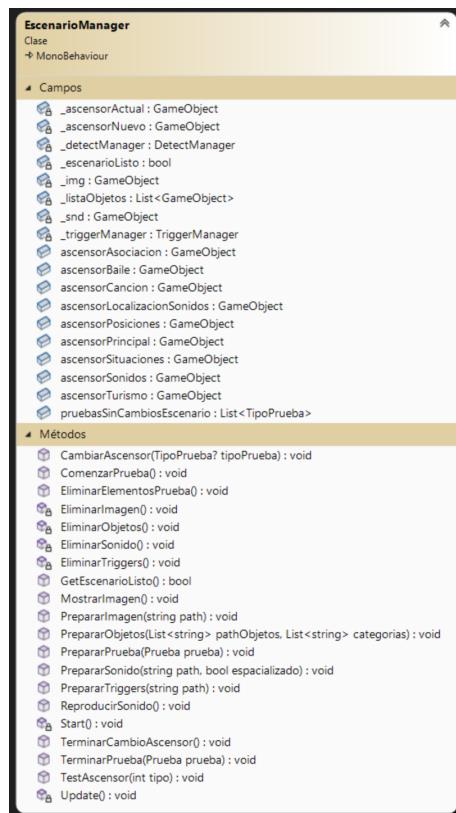


Figura 7.84: Diagrama de clase para EscenarioManager.

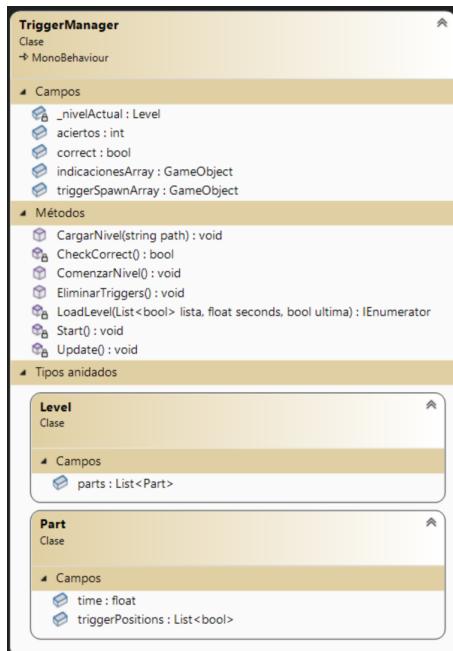


Figura 7.85: Diagrama de clase para TriggerManager.

en varias posiciones. Este funcionamiento se explica en más profundidad en la Iteración 2 de la entrega 2 (sección 7.2.3).

Detect Manager Este manager gestiona todas las zonas de detección del juego, las cuales se encuentran en las pruebas de asociación y la de sonidos. La función de estas zonas es detectar si los objetos que entran en contacto con ellas tienen una etiqueta determinada o no. Cuando comienza una prueba este manager localiza las zonas que hay y las asocia a dos categorías A y B (véase `_zonasCatA` y `_zonasCatB` en la figura 7.86), ya que los objetos en la prueba de asociación deben dividirse en dos grupos.

A su vez, cada zona de detección (DetectZone) tiene dos categorías asignadas, una como principal y otra como secundaria. Cada vez que un objeto entra en la zona se comprueba si pertenece a una de estas categorías y si es así, se marca la zona como correcta para dicha categoría y se lleva la cuenta del numero de objetos que hay dentro de la zona (véase figura 7.87).

En la prueba existen dos grupos de zonas de detección, pero el jugador puede elegir qué categoría agrupar a cada lado, por lo que es imprescindible que se pueda tener en cuenta todas las posibles situaciones en las que se puede completar la prueba, de eso se encarga el método `ComprobarCorrecto()` de la clase DetectManager como se muestra en la figura 7.88.

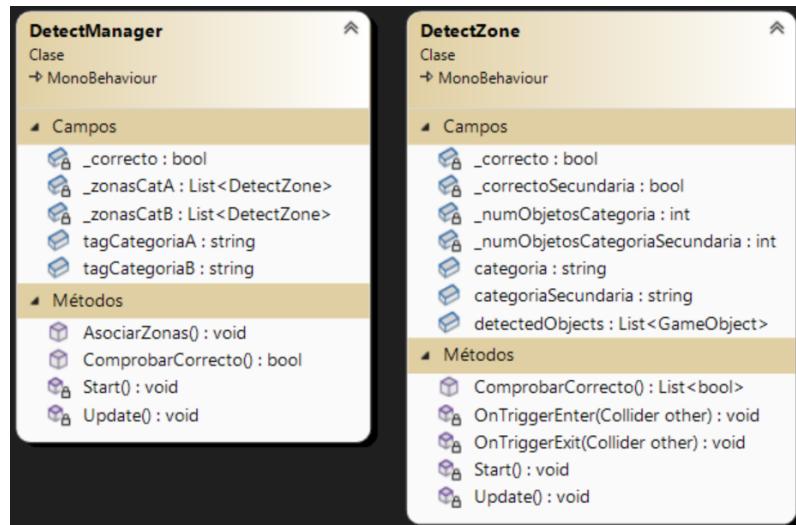


Figura 7.86: Diagrama de clases de DetectManager y DetectZone.

Ascensor Esta clase la tienen cada uno de los distintos ascensores que hay en el juego, uno por cada prueba. Su cometido es indicar si el ascensor se encuentra en su posición baja o elevada y hacer que transicione de una a otra de forma suave.

Para hacer dicho movimiento se utiliza un método asíncrono que en Unity se denominan *Coroutines* para hacer que en cada fotograma del juego avance ligeramente hacia su nueva posición. Cualquier movimiento asociado a la velocidad de fotogramas puede cambiar completamente cuando el juego se ejecuta a otra velocidad de la prevista, por ejemplo si el dispositivo de juego es más potente de lo esperado, o todo lo contrario, si el dispositivo no puede mantener una velocidad estable.

Para evitar este problema se utiliza el valor interno de Unity llamado 'Time.deltaTime' que contiene para cada fotograma el tiempo en milisegundos que ha pasado desde el anterior fotograma, multiplicando la velocidad de movimiento por este valor como se muestra en la figura 7.89 se puede conseguir que este movimiento sea independiente.

Prueba Esta clase contiene todos los elementos necesarios para los distintos tipos de pruebas. Se ha elegido crear una estructura de herencia, donde cada tipo de prueba tiene su propia clase que hereda de esta. Esto es porque cada prueba tiene sus propias características, como las imágenes a mostrar, los sonidos a reproducir, los objetos que mostrar y especialmente cada una tiene su propia condición de victoria. Por esto, cada clase hija sobrescribe los métodos

```
④ Mensaje de Unity | 0 referencias
private void OnTriggerEnter(Collider other)
{
    detectedObjects.Add(other.gameObject);
    if (other.gameObject.tag == categoria)
    {
        _correcto = true;
        _numObjetosCategoria++;
    }
    else if (other.gameObject.tag == categoriaSecundaria)
    {
        _correctoSecundaria = true;
        _numObjetosCategoriaSecundaria++;
    }
}

④ Mensaje de Unity | 0 referencias
private void OnTriggerExit(Collider other)
{
    detectedObjects.Remove(other.gameObject);
    if (other.gameObject.tag == categoria)
    {
        _numObjetosCategoria--;
        if (_numObjetosCategoria < 1)
        {
            _correcto = false;
        }
    }
    else if (other.gameObject.tag == categoriaSecundaria)
    {
        _numObjetosCategoriaSecundaria--;
        if (_numObjetosCategoriaSecundaria < 1)
        {
            _correctoSecundaria = false;
        }
    }
}
```

Figura 7.87: Métodos de la clase DetectZone para comprobar si los objetos que entran y salen son de la categoría adecuada.

```
2 referencias
public bool ComprobarCorrecto()
{
    List<bool> resultadoParcial = new List<bool>();
    bool correctoParcial = true;
    bool hayObjetoValidoA = false;
    bool hayObjetoValidoB = true;

    List<bool> resultado = null;
    for (int i = 0; i < _zonasCatA.Count; i++)
    {
        resultado = _zonasCatA[i].ComprobarCorrecto();
        if (i == 0)
        {
            resultado = _zonasCatA[i].ComprobarCorrecto();
            resultadoParcial = _zonasCatA[i].ComprobarCorrecto();
            hayObjetoValidoA = (resultado[0] || resultado[1]);
        }
        else if (resultado[0] != resultadoParcial[0] || resultado[1] != resultadoParcial[1])
        {
            correctoParcial = false;
        }
    }
    for (int i = 0; i < _zonasCatB.Count; i++)
    {
        //resultadoParcial = _zonasCatB[0].ComprobarCorrecto();
        resultado = _zonasCatB[i].ComprobarCorrecto();
        if (i == 0)
        {
            resultado = _zonasCatB[i].ComprobarCorrecto();
            resultadoParcial = _zonasCatB[i].ComprobarCorrecto();
            hayObjetoValidoB = (resultado[0] || resultado[1]);
        }
        else if (resultado[0] != resultadoParcial[0] || resultado[1] != resultadoParcial[1])
        {
            correctoParcial = false;
        }
    }

    correcto = correctoParcial && hayObjetoValidoA && hayObjetoValidoB;
}

return _correcto;
}
```

Figura 7.88: Método de la clase DetectManager para comprobar si las zonas de detección contienen los objetos correctos.

```
1 referencia
IEnumerator Bajar()
{
    isArriba = false;
    isAbajo = false;
    _enMovimiento = true;
    while (_enMovimiento)
    {
        float step = speed * Time.deltaTime;
        transform.position = Vector3.MoveTowards(transform.position, targetAbajo, step);

        if (Vector3.Distance(transform.position, targetAbajo) < 0.001f)
        {
            _enMovimiento = false;
            isAbajo = true;
        }
        yield return null;
    }
}
```

Figura 7.89: Método asíncrono que baja el ascensor en cada fotograma.

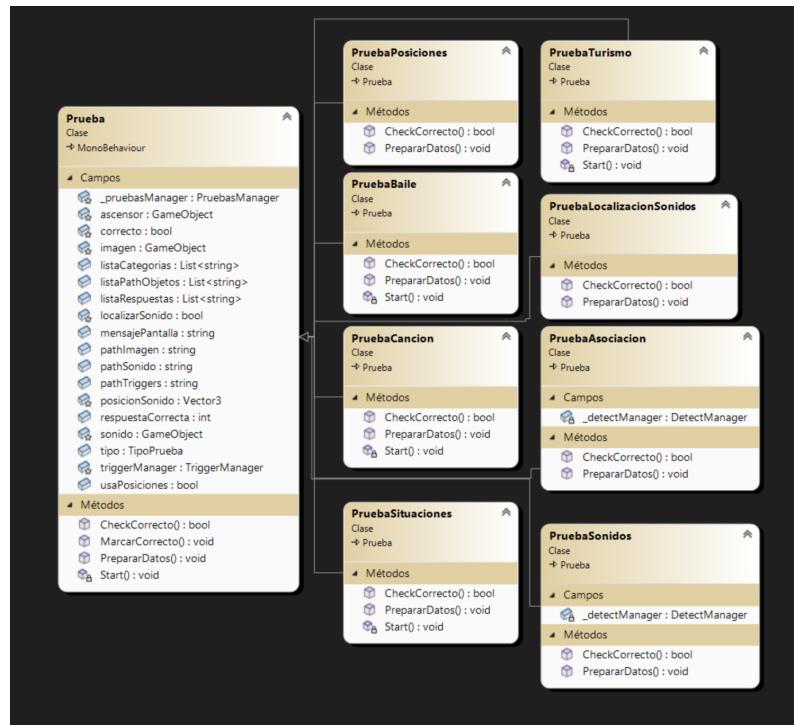


Figura 7.90: Diagrama de la clase Prueba y sus clases herederas.

```
2 referencias
public override bool CheckCorrecto()
{
    return _detectManager.ComprobarCorrecto();
}
```

Figura 7.91: Método sobrescrito para comprobar si la prueba de asociaciones es correcta.

PrepararDatos() y CheckCorrecto() (véase figura 7.90).

En la figura 7.91 se muestra el método sobrescrito con el que la clase PruebaAsociacion utiliza el manager de detección para comprobar si los objetos han sido colocados en el lugar correcto (figura 7.88).

Diagrama final de clases

A continuación, en la figura 7.92 se muestra el diagrama de clases completo desarrollado para este proyecto. Este diagrama ha sido generado de forma automática mediante Visual Studio 2022. En este diagrama no aparecen las líneas de unión entre clases, ya que a pesar de que existen referencias entre ellas, debido al funcionamiento de Unity, las referencias no son directas entre clases, si

no que son con objetos de tipo GameObject, que a su vez contiene dicha clase como uno de sus componentes. En Unity todos los elementos del juego son GameObjects, con distintos componentes que serían las clases. A pesar de esto, los nombres que se usan son claros para evitar confusiones respecto al tipo de clase que contienen.

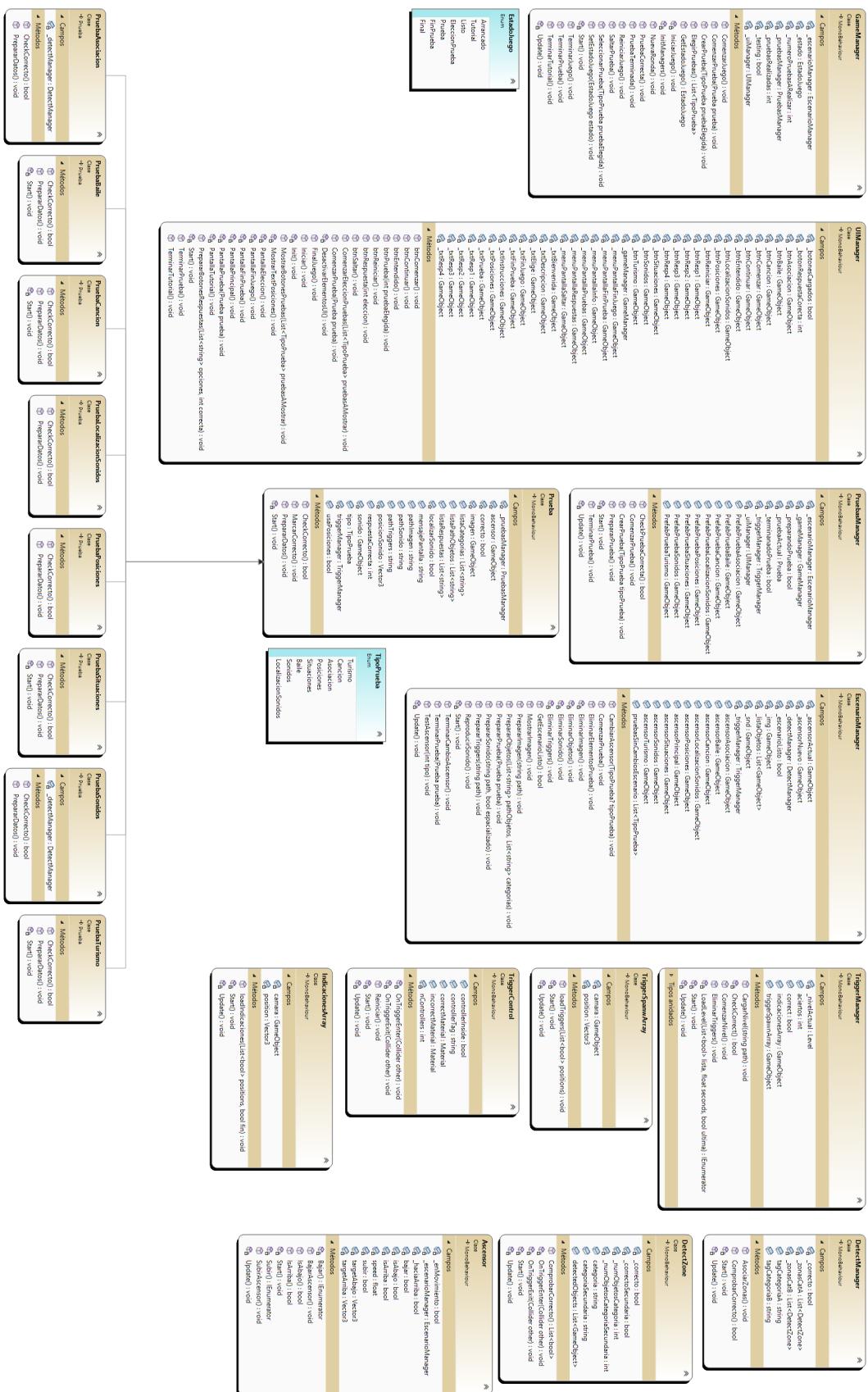


Figura 7.92: Diagrama de las clases desarrolladas para este proyecto.

Secuencia del juego

Para ilustrar el flujo y la secuencia de eventos que suceden durante una partida normal del juego, se ha creado un diagrama de secuencia (figura 7.93) en el que aparecen el usuario y las principales clases manager, mostrando las interacciones que realizan en el tiempo.

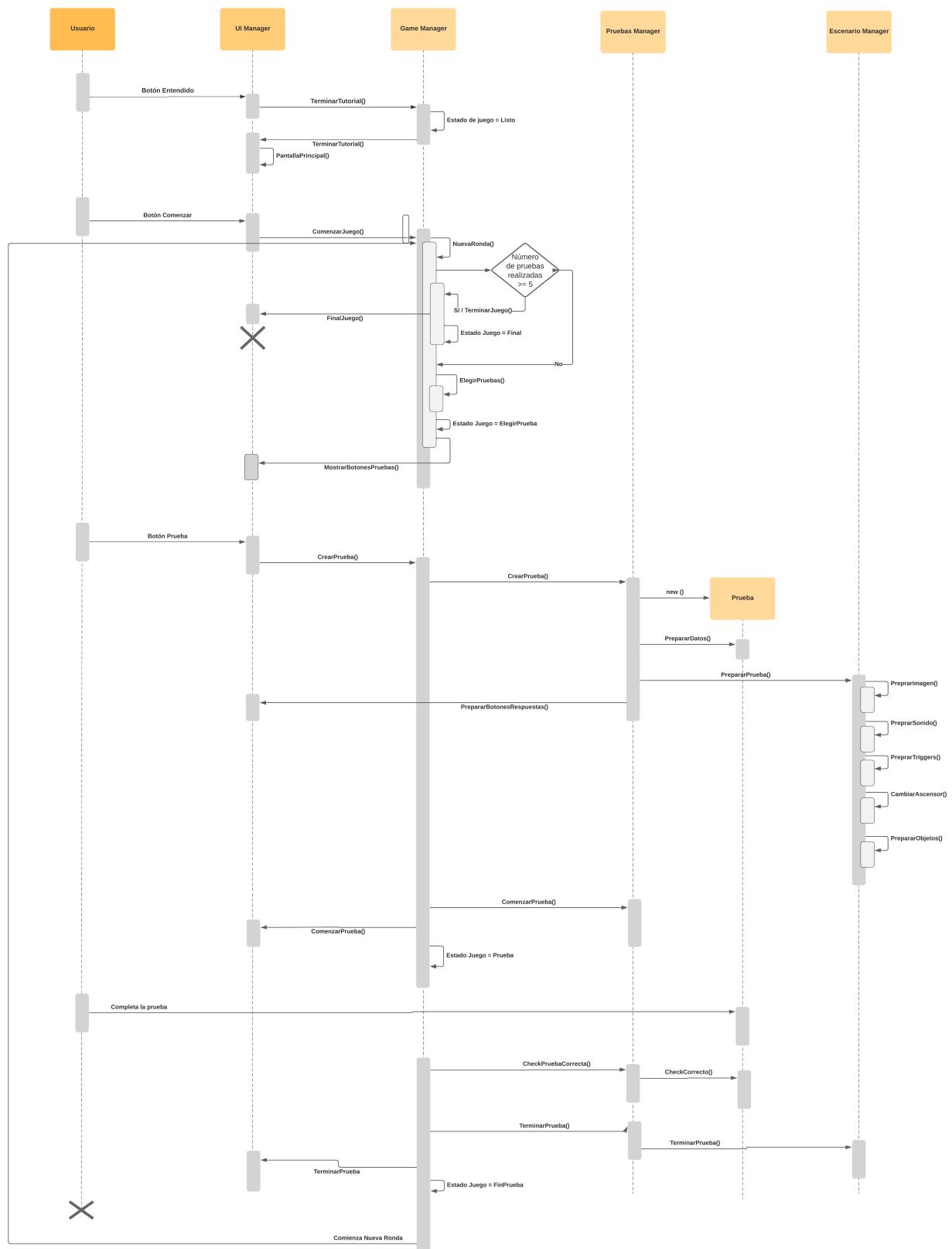


Figura 7.93: Diagrama de secuencia de una partida del juego.

7.5.3. Iteración 2

7.5.4. Conclusiones de la entrega 5

Parte IV

Conclusiones

Conclusiones y líneas futuras

Conclusiones y líneas futuras

Después de todo el desarrollo del proyecto, es pertinente hacer una valoración final del mismo, respecto a los resultados obtenidos, las expectativas o el resultado de la experiencia acumulada.

Esta sección es indispensable y en ella se ha de reflejar, lo más claramente posible, las aportaciones del trabajo con unas conclusiones finales.

Además, considerando también el estado de la técnica, se deben indicar las posibles líneas futuras de trabajo, proponer otros puntos de vista o cualquier otra sugerencia como postámbulo del presente trabajo, para ser considerada por el lector o el tribunal evaluador.

Parte V

Bibliografía

Bibliografía

- [1] Angela K. Troyer. *Serial Position Effect*, pages 2263–2264. Springer New York, New York, NY, 2011.
- [2] Esteve. Cuadernos de estimulación cognitiva. <https://www.esteveagora.com/farmaceutico/cuadernos-de-estimulacion-cognitiva>.
- [3] Rubio. 1 Memoria - Estimulación cognitiva. <https://cuadernos.rubio.net/estimulacion-cognitiva-memoria-1>.
- [4] Central Informativa. Actividades para adultos mayores de Hogar Geriátrico. <https://www.redadultomayor.org/actividades-para-adultos-mayores-de-hogar-geriatrico/>, 2016.
- [5] NeuronUP, estimulación cognitiva y neurorrehabilitación. <https://www.neuronup.com/es/>.
- [6] Neuroreality. Koji's Quest. <https://medkitvr.com/neuroreality-kojis-quest/>.
- [7] Virtuleap. Enhance. <https://virtuleap.com/enhance>.
- [8] Guillaume de Lorris. Roman de la Rose. M. 948, fol 12r., 1240.
- [9] Leonardo Da Vinci. La última cena. Temple y óleo sobre yeso, 1498.
- [10] Pietro Perugino. Entrega de las llaves a San Pedro. Fresco - Capilla Sixtina, 1482.
- [11] Robert Barker. Un vistazo a Londres y Westminster, 1792.
- [12] Robert Mitchell. Plans, and views in perspective, with descriptions of buildings erected in England and Scotland; and ... an essay to elucidate the Grecian, Roman and Gothic Architecture. (Plans, descriptions et vues en perspective, etc.). Grabado al aguatinta, 15 de Mayo 1801.

- [13] The Popular science monthly vol. 21, pág. 47. <https://archive.org/details/popularsciencemo21newy/page/n3/mode/2up>, 1882.
- [14] Morton Heilig. Illustration of Morton Heilig's Sensorama device, precursor to later virtual reality systems. https://commons.wikimedia.org/wiki/File:Sensorama_patent_fig5.png.
- [15] igreet.co. Brief History of Augmented Reality. <http://www.igreet.co/brief-history-of-augmented-reality/>.
- [16] Ivan Sutherland. The Sword of Damocles - Head Mounted Display. <http://www.dsource.in/course/virtual-reality-introduction/evolution-vr/sword-damocles-head-mounted-display>, 1963.
- [17] Dave Pape. A VPL Research Eyephone head mounted display and Dataglove, with Polhemus tracking system. The first commercially marketed virtual reality system. Displayed at the Nissho Iwai showroom in Tokyo. https://commons.wikimedia.org/wiki/File:VPL_Eyephone_and_Dataglove.jpg, 1999.
- [18] Evan Amos. A North American Virtual Boy game console. <https://commons.wikimedia.org/wiki/File:Virtual-Boy-Set.png>, 2011.
- [19] Nintendo R&D1. Screenshot showing typical gameplay from the game Mario's Tennis for the Virtual Boy. https://en.wikipedia.org/wiki/File:VB_Mario%27s_Tennis.png, 1995.
- [20] Alexey Severin. Product visualization: Oculus Rift DK1. <https://www.severin3d.com/oculus-rift-dk1>.
- [21] Imagen promocional de Sony. <https://www.playstation.com/es-es/explore/playstation-vr/>.
- [22] othree. Google Cardboard. <https://www.flickr.com/photos/othree/14519574116/>, 2014.
- [23] Jesús Maturana. HTC Vive, análisis: esto sí que es realidad virtual interactiva. <https://www.xataka.com/analisis/htc-vive-analisis-esto-si-que-es-realidad-virtual-interactiva>, 2017.
- [24] Valve Corporation. Imagen promocional de Valve Index. <https://www.valvesoftware.com/es/index>.

- [25] Página de producto para Meta Quest 2 en Amazon.es. <https://www.amazon.es/Oculus-Quest-realidad-virtual-avanzada/dp/B08QDQK9GL>.
- [26] Unity. Cómo financiar y promocionar tu juego. <https://unity3d.com/es/how-to/finance-and-promote-your-game>.
- [27] Colin Campbell. Monument Valley 2 review. <https://www.polygon.com/2017/6/8/15762978/monument-valley-2-review-ios-ipad-iphone>, 2017.
- [28] Documentación Unreal Engine 4. Tools and Editors. Level Editor. <https://docs.unrealengine.com/en-US/GettingStarted/SubEditors/index.html>.
- [29] esportshound.com. Fortnite. <http://www.esportshound.com/fortnite-2/>.
- [30] YoYo Games Ltd. GameMaker Studio 2 Web. https://store.steampowered.com/app/585600/GameMaker_Studio_2_Web/?l=spanish, 2017.
- [31] Butterscotch Shenanigans. Crashlands. <https://store.steampowered.com/app/391730/Crashlands/?l=spanish>, 2016.
- [32] Fabio Alessandrelli. Godot Editor running in a web browser. <https://godotengine.org/article/godot-editor-running-web-browser>, 2020.
- [33] igbd.com. RivenTails Defense Press kit. <https://www.igdb.com/games/riventails-defense/presskit>.
- [34] panoramaaudiovisual.com. Despliegue audiovisual escenográfico en 'Atrapa un millón'. <https://www.panoramaaudiovisual.com/2011/11/07/despliegue-audiovisual-escenografico-en-atrapa-un-millon/>, 2011.
- [35] Mat Paget. Oculus Quest Sold Out At Major Retailers. <https://www.gamespot.com/articles/oculus-quest-sold-out-at-major-retailers/1100-6476020/>, 2020.
- [36] Marlady Ortiz. Modelo iterativo. <http://isw-udistrital.blogspot.com/2012/09/ingenieria-de-software-continuacion.html>.
- [37] Roger S. Pressman. *Ingeniería del software: Un enfoque práctico*, page 28. Mc Graw Hill, 2005.
- [38] Cris Busquets. Medir la usabilidad con el Sistema de Escalas de Usabilidad (SUS). <https://www.uifrommars.com/como-medir-usabilidad-que-es-sus/>.

- [39] antena3.com. ¿Está Will Smith? Las sorpresas entre el público que regresa al plató de '¡Ahora caigo!'. <https://www.antena3.com/programas/ahora-caigo/momentos/esta-will-smith-las-sorpresa-entre-el-publico-que-regresa-al-plato-de-ahora-caigo-20200615ed49ce9d84fea00015aad17.html>, 2020.
- [40] Sebastián Salgado González. La filosofía de Aristóteles. <http://guindo.pntic.mec.es/ssag0007/filosofica/aristoteles-duereras.pdf>, 2012.
- [41] Alan Kim. Wilhelm Maximilian Wundt. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2022 edition, 2022.
- [42] Hermann Ebbinghaus. Chapter iii: The method of investigation. In *Memory: A Contribution to Experimental Psychology*. New York by Teachers College, Columbia University, 1885.
- [43] Magali Bovet. Piaget's theory of cognitive development and individual differences, 1976.
- [44] Ronald C. Petersen, Glenn E. Smith, Stephen C. Waring, Robert J. Ivnik, Eric G. Tangalos, and Emre Kokmen. Mild Cognitive Impairment: Clinical Characterization and Outcome. *Archives of Neurology*, 56(3):303–308, 03 1999.
- [45] Huey ed, manly jj, tang mx, et al. course and etiology of dysexecutive mci in a community sample. *alzheimers dement*. 2013;9(6):632-639. doi:10.1016/j.jalz.2012.10.014. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3933297/>.
- [46] Moheb Costandi. Neuroplasticidad humana. In *Neuroplasticity*. The MIT Press, Cambridge, Massachusetts, 2016, 2016.
- [47] David Moreau, Brooke N Macnamara, and Zach Hambrick. Overstating the role of environmental factors in success: A cautionary note, Jul 2018.
- [48] Esteve. Cuadernos de ejercicios de estimulación cognitiva. <https://www.esteve.com/es/pacientes/demencia>.
- [49] grupociudadjardin. Juegos para personas mayores en residencias. <https://www.grupociudadjardin.com/ludoterapia-para-ancianos/>, Oct 2022.
- [50] Lumosity. Descubre de lo que es capaz tu mente. <https://www.lumosity.com/es/>.

- [51] NeuronUP. Centros de neurorehabilitación. <https://www.neuronup.com/>.
- [52] NeuroReality. Science-based cognitive training in virtual reality. <https://neuro-reality.com/>.
- [53] Virtuleap. Brain training vr app. <https://virtuleap.com/>.
- [54] Christian Moro, James Birt, Zane Stromberga, Charlotte Phelps, Justin Clark, Paul Glasziou, and Anna Mae Scott. Virtual and augmented reality enhancements to medical and science student physiology and anatomy test performance: A systematic review and meta-analysis. *Anatomical Sciences Education*, 14(3):368–376, May 2021. This article is protected by copyright. All rights reserved.
- [55] Markman Ellis. The spectacle of the panorama. <https://www.bl.uk/picturing-places/articles/the-spectacle-of-the-panorama>.
- [56] Charles Wheatstone. Contributions to the physiology of vision.—part the first. on some remarkable, and hitherto unobserved, phenomena of binocular vision. *Philosophical Transactions*, 128:371 – 394, Jun 1838.
- [57] Howard Rheingold. *Virtual Reality*, pages 105 –113. New York: Simon Schuster, 1992.
- [58] Aldo Faisal. Computer science: Visionary of virtual reality. *Nature*, 551:298–299, Nov 2017.
- [59] Malachai Jacobs. Here's what you didn't know about the history of virtual reality. <https://delta2020.com/blog/221-here-s-what-you-didn-t-know-about-the-history-of-virtual-reality>.
- [60] Ben Delaney. *Sex, Drugs and Tessellation: The Truth About Virtual Reality, as Revealed in the Pages of CyberEdge Journal*, page 274. CreateSpace Independent Publishing Platform, 2014.
- [61] Steven Boyer. A virtual failure: Evaluating the success of nintendos virtual boy. *Velvet Light Trap*, 64:23–33, 2009.
- [62] Tim Stevens. Nintendo virtual boy review. <https://www.engadget.com/2011-03-21-nintendo-virtual-boy-review.html>.
- [63] Kevin Mellott. Oculus rift development kit. <https://mellottsvrpage.com/index.php/oculus-rift-development-kit/>.

- [64] Leo Kelion. Htc reveals virtual reality headset with valve at mwc. <https://www.bbc.com/news/technology-31664948>.
- [65] Martin Armstrong. Meta leads the way in vr headsets. <https://www.statista.com/chart/29398/vr-headset-kpis/>.
- [66] Valve index headset specifications. <https://www.valvesoftware.com/en/index/headset>.
- [67] Will Greenwald. Meta quest 2 review. <https://www.pcmag.com/reviews/oculus-quest-2>.
- [68] What is a gaming or game engine? <https://www.arm.com/glossary/gaming-engines>.
- [69] Arnia. What makes unity so popular in game development? <https://www.arnia.com/what-makes-unity-so-popular-in-game-development/>.
- [70] Compare plans - unity. <https://unity.com/compare-plans>.
- [71] Jeff Drake. 24 great games that use the unreal 4 game engine. <https://www.thegamer.com/great-games-use-unreal-4-game-engine/>.
- [72] Frequently Asked Questions (FAQs) - Unreal Engine. <https://www.unrealengine.com/en-US/faq>.
- [73] Funciones de GameMaker. <https://gamemaker.io/es/gamemaker/features>.
- [74] Planes de pago de GameMaker. <https://gamemaker.io/es/get#comparison>.
- [75] Why godot is right for you. <https://godotengine.org/features/>.
- [76] Manuel A. Barajas Bustillos; Rosa María Reyes M; Jorge de la Riva R.; Aidé Maldonado Macías. Estudio comparativo de cuestionarios para la evaluación de la usabilidad en software. *Revista Ingeniantes. Año 4. No 2, 1:3–7, 2017.*
- [77] Create Your First VR App on Meta Quest Headset. <https://developer.oculus.com/documentation/unity/unity-tutorial-hello-vr/#step-4-import-integration-sdk-from-oculus-developer-center>.

Parte VI

Apéndices

Apéndice A

Game Desing Document

Esta es una versión reducida y adaptada a este trabajo de un GDD. Un documento de este tipo a nivel comercial abarca un mayor ámbito y se sale del objetivo de este proyecto.

A.1. Información general

Este es un juego serio para el entrenamiento cognitivo de los jugadores. Está enfocado a personas que debido a la edad o algún tipo de accidente o enfermedad, han visto reducidas sus capacidades cognitivas, el objetivo es que la persona que lo juegue vea mejoradas o restauradas dichas capacidades a lo largo de varias sesiones de juego. Se trata de un juego que utiliza la realidad virtual como medio de potenciar la efectividad de los ejercicios clásicos de entrenamiento cognitivo. En este juego, el jugador se ve inmerso en un escenario similar al de un concurso de TV en el que se le presentan distintas pruebas que debe superar. Esto son ejercicios cognitivos adaptados para sacar el máximo partido a la RV, buscando tanto la eficacia como la diversión para el jugador.

A.1.1. Resumen del juego

El jugador se ve envuelto en un concurso de TV donde el escenario irán apareciendo y desapareciendo distintos decorados y objetos con los que interactuar, y que pondrán a prueba sus capacidades cognitivas en cada minijuego.

A.1.2. Objetivos a alcanzar por el juego

En este juego no se busca que el jugador alcance un estado de victoria o llegue a una derrota, se trata de un juego serio para entrenar la cognición. Al final de cada prueba se otorgan unos puntos basados ligeramente en la habilidad del jugador, cuya única función es la de motivarlo.

A.1.3. Justificación del juego

La importancia de este juego reside en la adaptación a las nuevas tecnologías de los ejercicios clásicos de entrenamiento cognitivo. Mediante la gamificación de estos entrenamientos se intenta aumentar la motivación y diversión de los pacientes que lo utilicen para hacer más llevadera la labor. De igual forma, la realidad virtual permite además de aumentar el interés del jugador, incrementar la eficacia de cada uno de los juegos, buscando un tratamiento y recuperación más rápidos.

A.1.4. Core gameplay

Después de comenzar, al jugador se le presentará la opción de elegir una prueba entre las disponibles, a continuación tendrá que superar la prueba elegida. Después de terminar la prueba, se repite el bucle de elegir prueba y superarla hasta que se completen 10 pruebas.

A.1.5. Características del juego

El juego transcurre en un único escenario que se ve modificado y adaptado para cada prueba. No existen otros personajes a parte del jugador y el único reto que se plantea es el de superar las distintas pruebas.

Género

El juego puede clasificarse con una mezcla de juego de preguntas y compilación de minijuegos. Ejemplos de estos estilos son: Buzz y WarioWare respectivamente.

Número de jugadores

El juego es para un solo jugador.

Público objetivo

Principalmente para personas a partir de 60 años o que sufran de deterioro cognitivo leve, así como cualquier otra persona que quiera entrenar sus habilidades cognitivas.

Plataformas de destino

El juego está destinado a dispositivos de realidad virtual, principalmente Meta Quest 2.

Estética y arte del juego. Estilo visual

El juego se ambienta en un plató de televisión. El plató está completamente equipado con el decorado necesario para realizar un concurso y son visibles también elementos que normalmente no lo serían como los foco o el público. Por las limitaciones del proyecto, los gráficos son sencillos y poco detallados.

Resumen de historia

Este juego no tiene historia principal, transcurre en un concurso de TV donde se presenta a jugador con pruebas que superar.

A.1.6. Características del jugador

Personas que deseen realizar entrenamientos mentales o que sufran deterioro cognitivo leve y deban realizar rehabilitación cognitiva pueden encontrar en este juego una forma más divertida de realizarlo.

A.1.7. Recursos iniciales

Este juego es desarrollado por una única persona y sin presupuesto inicial, con un tiempo previsto de 3 meses.

A.2. Mecánicas

Para evitar alargar este documento y repetir información, las mecánicas principales del juego pueden encontrarse en la sección **Pruebas** (3.1).

A.2.1. Elementos de juego

Los elementos que aparecen en este juego pueden dividirse en las categorías de: Escenario, que engloba a todos los objetos pertenecientes al decorado cuya única función es ambientar (focos, grada, escenario). Los interactuables, son aquellos con los que el jugador interacciona directamente, especialmente para progresar en el flujo del juego (botones para elegir prueba, pantalla del escenario y toda la interfaz). Finalmente, los objetos de prueba, son aquellos elementos que aparecen y desaparecen para cada prueba concreta, sirven como elemento principal de la prueba a resolver (tocadiscos, avión, imágenes de memoria).

A.2.2. Reglas

El jugador podrá moverse en una zona limitada, pulsar botones, coger y soltar objetos de prueba y utilizar la pantalla del escenario.

Inteligencia artificial

Este juego no dispone de ningún tipo de inteligencia artificial.

A.2.3. Elementos de juego: Mundo

Las principales zonas del mundo del juego son el centro del escenario donde pasará la mayor parte del tiempo el jugador; la mesa de juegos, que es una zona que se eleva del suelo para ofrecer una mesa llena con los elemento necesarios para que usuario realice una prueba; y la zona de baile, una zona situada junto al escenario principal en la que se llevan a cabo las pruebas de movimiento.

A.2.4. Elementos de registro y progreso

Este juego no guarda el progreso, pero podrían crearse perfiles donde ver la evolución de resultados del jugador.

A.3. Dinámica

El juego comienza con el jugador apareciendo en el centro de un plató de televisión. A través de una pantalla gigante se le introduce al juego y se le pone en situación. A continuación, desde esa misma pantalla gigante con la que puede interactuar a modo de interfaz, se le presentará con dos tipos de pruebas diferentes. El jugador elegirá señalando la opción deseada y dará comienzo el cambio de escenario. Dependiendo del tipo de prueba, el jugador puede ser transportado a una nueva zona o mantenerse en la misma, y se le presentará ascendiendo desde el suelo una mesa con los elementos de la prueba.

Tras realizar la prueba, el jugador recibe su recompensa y se procede a restablecer el escenario a su forma original. Finalmente, en la pantalla vuelve a presentarse la elección para una nueva prueba. Este proceso se repite hasta haber completado el número de pruebas deseado.

Apéndice B

Cuestionarios

Reescribir preguntas formularios. Rellenar respuestas cuestionarios

B.1. Plantilla

Este cuestionario consta de 10 preguntas para evaluar la eficacia, eficiencia y satisfacción del sistema que acabas de probar.

1. Creo que me gustaría utilizar este sistema con frecuencia.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

2. Encontré el sistema innecesariamente complejo.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

3. Pensé que el sistema era fácil de usar.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

4. Creo que necesitaría el apoyo de un técnico para poder utilizar este sistema.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

5. Encontré que las diversas funciones de este sistema estaban bien integradas.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

6. Pensé que había demasiada inconsistencia en este sistema.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

7. Me imagino que la mayoría de la gente aprendería a utilizar este sistema muy rápidamente.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

8. Encontré el sistema muy complicado de usar.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

9. Me sentí muy seguro usando el sistema.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

10. Necesitaba aprender muchas cosas antes de empezar con este sistema.

Totalmente en desacuerdo	En desacuerdo	Neutro	De acuerdo	Totalmente de acuerdo

B.2. Resultados usabilidad

Tras presentar este cuestionario a X personas, se han obtenido los resultados de usabilidad mostrados en la tabla B.1. Obteniendo una puntuación media en la escala de usabilidad de X. Véase sección 5.3 para más información sobre esta puntuación.

Encuestado	Preg. 1	Preg. 2	Preg. 3	Preg. 4	Preg. 5	Preg. 6	Preg. 7	Preg. 8	Preg. 9	Preg. 10	Suma impares	Suma pares	Total SUS
A	4	3	4	4	4	2	4	2	2	3	18	14	60
B	5	1	5	4	5	2	5	1	5	2	25	10	87.5
C	4	1	4	3	4	2	4	2	3	2	19	10	72.5
D	4	1	5	2	4	2	4	2	4	1	21	8	82.5
E	4	2	4	4	4	2	5	2	3	3	20	13	67.5
F	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	0	0	0	0	0	0	0

Tabla B.1: Resultados de usabilidad obtenidos tras la prueba

