



Universidad
Zaragoza

Trabajo Fin de Grado

Desarrollo de una aplicación en realidad mixta para
el estudio de mecanismos de atención en entornos
inmersivos

Development of a mixed reality application for
studying attentional mechanisms in immersive
environments

Autor

Jaime Bielsa Uche

Directores

Ana Serrano Pacheu

Daniel Martín Serrano

ESCUELA DE INGENIERÍA Y ARQUITECTURA
2024

AGRADECIMIENTOS

Este Trabajo de Fin de Grado es el punto de inflexión entre el Grado de Ingeniería Informática y el Máster de Gráfica, Robótica y Visión por Computador. Quiero agradecer a todo el equipo docente de la **Universidad de Zaragoza** por el magisterio recibido y por recibir, y la apuesta por una educación pública de calidad.

Agradecer también a mis compañer@s de clase, así como personal relacionado con el centro, por crear un ambiente cómodo para el estudio y la convivencia. La etapa universitaria ha estado llena de altibajos, estrés pero también buenos momentos que guardaré para toda la vida.

Agradecer, para la realización de este TFG, primero, a todas las personas que se han presentado voluntarias para probar la aplicación y aportar ideas. Luego, al **Graphics and Imaging Lab** como equipo de divulgación pero también de convivencia. Finalmente, gracias en especial a **Daniel** y **Ana** por su orientación, complicidad y profesionalidad a lo largo del proyecto.

En el marco personal, agradecer a mi familia y amistades por el cariño y la compañía cuando más la he necesitado.

Desarrollo de una aplicación en realidad mixta para el estudio de mecanismos de atención en entornos inmersivos

RESUMEN

La realidad extendida (XR) es un campo que ha ganado relevancia durante la última década debido a su potencial para crear experiencias inmersivas, teniendo aplicaciones en sectores como la educación, la medicina, los videojuegos o la robótica. Para crear experiencias realistas y atractivas, y poder optimizar los recursos disponibles, es importante comprender el comportamiento humano, y más particularmente la atención. La atención es un proceso que permite discretizar y seleccionar la información que recibe nuestro cerebro, el cual integra estímulos provenientes de múltiples sentidos, es decir, multimodales. No obstante, el estudio y modelado de la atención presenta desafíos.

El objetivo de este Trabajo Fin de Grado es explorar y comparar distintos mecanismos atencionales en realidad mixta (MR), combinando estímulos provenientes de distintos sentidos (visual, auditivo, y háptico). Para ello, se ha desarrollado una experiencia inmersiva en MR utilizando las gafas de XR Meta Quest 3, en la que los usuarios deben realizar búsquedas de objetos en entornos controlados haciendo uso de una serie de ayudas multimodales, que buscan evaluar cómo afectan diversos mecanismos de redirección de la atención a través de distintos sentidos a la capacidad de búsqueda de las personas.

Después, se ha realizado un estudio de usuario para validar la aplicación y analizar el impacto de dichos mecanismos atencionales. Los resultados muestran como los mecanismos de redirección permiten a los jugadores encontrar más rápido los elementos buscados, donde las ayudas sonoras, seguidas de las hápticas, ofrecen los mejores resultados, reforzando la importancia del uso de mecanismos atencionales y abriendo paso a futuros estudios sobre otros métodos de redirección de la atención.

Índice

1. Introducción	1
1.1. Contexto y motivación	1
1.2. Objetivos	3
1.3. Herramientas	4
1.3.1. Hardware	4
1.3.2. Software	4
1.3.3. Lenguajes de programación	4
1.3.4. Control de versiones	5
1.4. Planificación	5
2. Diseño e implementación	7
2.1. Herramientas clave	7
2.1.1. Meta Quest 3	7
2.1.2. Unity	8
2.1.3. Integración	10
2.2. Diseño de la experiencia	11
2.2.1. Descripción de la experiencia	11
2.2.2. Obtención de métricas	13
2.2.3. Navegación	14
2.2.4. Objetos	16
2.2.5. Interacción	16
2.3. Implementación de la aplicación	18
2.3.1. Lógica de la aplicación	18
2.3.2. Estados de la aplicación	18
2.3.3. Interfaz gráfica	19
2.3.4. Menús	20
2.3.5. Logger	20
2.3.6. Partida	20
2.3.7. Objetos virtuales	22

3. Pruebas y análisis	25
3.1. Pruebas a realizar	25
3.1.1. Descripción de las pruebas	25
3.1.2. Realización de las pruebas	26
3.2. Obtención de métricas	26
3.3. Desarrollo de ejecuciones	26
3.4. Análisis de resultados	28
3.4.1. Eficacia de las ayudas recibidas	28
3.4.2. Curva de aprendizaje	29
3.4.3. Curvas de aprendizaje según el modo	30
3.4.4. Dificultad de los objetos	31
3.4.5. Centrado del objetivo en la pantalla	32
3.5. Opiniones de los usuarios	32
3.5.1. Potencia de vibración en el modelo háptico	32
3.5.2. Muebles sin oclusión y como interceptores de rayos	33
4. Conclusiones	35
4.1. Limitaciones y trabajo futuro	36
5. Bibliografía	39
Lista de Figuras	41
Lista de Tablas	43
Anexos	44
A. Métricas de la prueba con usuarios	47

Capítulo 1

Introducción

1.1. Contexto y motivación

Se conoce la realidad extendida (o XR por sus siglas en inglés) como una tecnología que busca combinar el mundo real con un mundo virtual apegado a la realidad y que interactúa con esta. Es un término paraguas que engloba otros términos más frecuentes, como la realidad virtual (VR), que simula una realidad alterna evitando la realidad; la realidad aumentada (AR), que añade contenido virtual sobre un espacio real; y la realidad mixta (MR) que ofrece experiencias más inmersivas haciendo que la capa virtual interactúe con la real.

Los *Head-Mounted Displays* o HMD son dispositivos que ofrecen contenido visual a uno o los dos ojos del usuario, con apoyo de un soporte para la cabeza. A día de hoy, los HMD más utilizados para realizar experiencias en XR son las comúnmente llamadas gafas de XR, con capacidad de reproducir entornos XR detallados gracias a motores gráficos y que sustituyen la visión de los dos ojos del usuario, siendo algunos ejemplos de ellas las Apple Vision Pro, las HTC Vive Focus o las Meta Quest 3. Se puede comprobar el aspecto de estas últimas en la Figura 1.1.

La XR nos permite navegar en entornos en 360°, simulando la visión de una persona en el mundo real. Esto abre las puertas a una nueva serie de desafíos entre los que

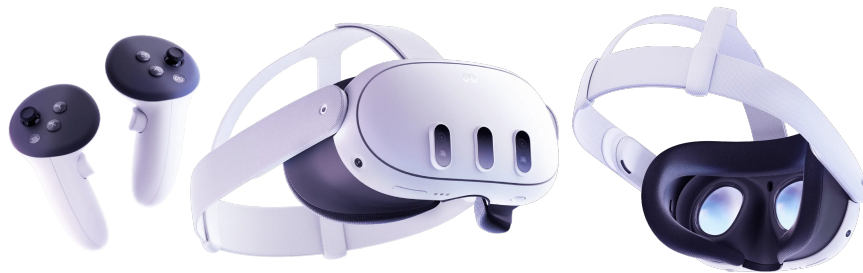


Figura 1.1: Controles izquierdo y derecho, parte frontal y parte trasera de las gafas de XR Meta Quest 3

se encuentran la gestión de la cámara para generar una experiencia adecuada, las disparidades de visión entre distintas personas, o la pérdida de contenido por no llegar siquiera a verlo, entre muchos.

Para generar experiencias satisfactorias es importante conocer cómo las personas responden a entornos simulados. La respuesta humana ante la presencia de estímulos es compleja de determinar. Suelen actuar procesos del cerebro que discretizan y sintetizan los estímulos recibidos para recibir solo aquella información que es relevante. Es lo que se conoce como la atención, un proceso cognitivo que consiste en la concentración selectiva de la información. El campo visual del ser humano es mucho más limitado que los 360° que ofrecen unas gafas de XR, siendo tarea del individuo decidir dónde mirar, es decir, a dónde dirigir la atención.

Además, la atención es un fenómeno que también actúa sobre canales sensoriales distintos al visual: el sonoro o el háptico. Ejemplos de esto son escuchar un objeto por el lateral, por detrás, sentir el tacto de un objeto, frío, calor,... La combinación de varios de estos canales para dirigir la atención del usuario es lo que se conoce como atención multimodal, otro de los desafíos de la XR.

Comprender y modelar la atención en espacios inmersivos es altamente relevante en varios frentes de la XR. Por ejemplo, sirve para mejorar la generación de contenido, optimizando la distribución de los objetos en el entorno para guiar la atención del usuario. También cobra importancia en otros aspectos como el *streaming* o la compresión, priorizando las áreas de la imagen con más saliencia, algo crucial a la hora de generar imágenes en tiempo real, un proceso de alto coste computacional. Estos conocimientos podrían incluso influir en el diseño de hardware, con la intención de reducir costes.

En base al paradigma que presenta la atención, hay muchos trabajos que han intentado modelar o predecir la atención visual. Otros trabajos, como [1], [2] y [3] han centrado sus esfuerzos en la atención multimodal, añadiendo estímulos sonoros. Para visualizar la atención, se recurre principalmente a dos representaciones. Un *scanpath* representa el recorrido de la mirada del usuario sobre el entorno, lo que se consigue combinando la posición de las gafas con el seguimiento de la mirada, o *eye tracking*. Si se quiere analizar el comportamiento de varias personas, se requiere de otra representación capaz de agregar todos sus *scanpaths*. Esta agregación genera *mapas de saliencia*, representaciones en un estilo de mapa de calor de cuánta atención han prestado los usuarios a las distintas zonas del entorno. Se pueden encontrar ejemplos de estas representaciones en la Figura 1.2.

La premisa de las XR de mantener una odometría controlada posibilita su uso para construir modelos de atención humana. Por otro lado, la naturaleza de la XR la

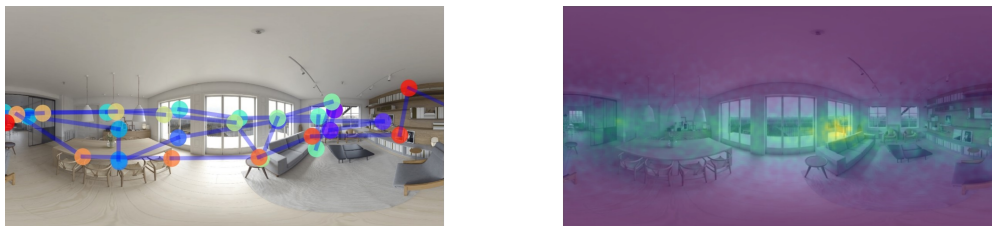


Figura 1.2: *Izda.*: ejemplo de *scanpath* sobre un entorno 360°. Los puntos se conectan para definir una trayectoria de atención visual. *Dcha.*: ejemplo de mapa de saliencia de un entorno 360°. Los áreas con colores más calientes son más visitados por los usuarios.

convierte en una herramienta atractiva para diseñar experiencias amenas con el usuario. Son el entorno elegido para construir muchas aplicaciones inmersivas, la gran mayoría en VR. Trabajar, concretamente con MR, añade la seguridad de jugar sobre el entorno real, pudiendo interactuar de la misma forma con la capa virtual.

De esta forma, este Trabajo Fin de Grado tiene como objetivo conocer las posibilidades de las gafas de XR Meta Quest 3 para crear una experiencia inmersiva en MR que integre elementos virtuales sobre el mundo real. La propuesta es el desarrollo de una experiencia interactiva, de carácter divulgativo y exploratorio, en la que los usuarios deban encontrar una serie de objetos, utilizando mecanismos de guiado de la atención (visuales, sonoros y/o hápticos). La experiencia servirá como herramienta de divulgación de conceptos relacionados con la atención multimodal y, por otro, para evaluar las capacidades de las Meta Quest 3 en este tipo de aplicaciones.

1.2. Objetivos

El alcance del Trabajo de Fin de Grado incluye lo enumerado a continuación:

- Estudio del estado del arte en MR y conceptos como atención y multimodalidad (Apartado 1.1).
- Toma de contacto con las gafas de XR Meta Quest 3 y comprender sus capacidades y limitaciones; y toma de contacto con el motor de videojuegos Unity y su integración con las Quest (Apartado 2.1).
- Diseño de la experiencia, procurando dar cabida a la atención multimodal y con intención de recopilar los resultados de las partidas (Apartado 2.2).
- Implementación de la aplicación en Unity (Apartado 2.3).
- Análisis del desempeño de personas voluntarias en la experiencia, y estudio de la eficacia de los estímulos recibidos en base a su desempeño (Capítulo 3).

1.3. Herramientas

A continuación, se mencionan todas las herramientas que se han empleado para realizar el proyecto:

1.3.1. Hardware

- Las gafas de XR que se van a utilizar son las **Meta Quest 3**¹, lanzadas al mercado hace poco más de un año de la escritura de esta memoria. Cuentan con un campo de visión de 110°, una resolución de 2064 × 2208 píxeles por ojo y una tasa de resolución de 120 fotogramas por segundo. Además, dispone de 18 cámaras que permiten a las gafas funcionar con *passthrough*, es decir, visualizar el mundo real con las gafas (lo que posibilita experiencias de AR y MR). Se habla más en detalle de las funcionalidades de las gafas en el Subapartado 2.1.1. Las MQ3 no requieren de dispositivos o procesamiento externos para funcionar. Además, son capaces de escanear, personalizar y guardar habitaciones (paredes, puertas, ventanas, mesas y otros muebles) para luego utilizar en las aplicaciones.
- Para programar la aplicación se utiliza un ordenador personal con características cercanas a la gama alta: una CPU Intel Core i7 de 10ª generación, una GPU Nvidia GeForce 1660Ti, 16 GB de RAM y 1 TB de memoria SSD.

1.3.2. Software

- Para implementar la experiencia se utiliza el motor Unity, que proporciona un contexto de elementos que se inician y, a continuación, se van actualizando al ritmo de la tasa de refresco de imagen. Tiene muchas extensiones y colecciones de objetos, materiales, etc. mantenidas por la comunidad y algunos organismos. Es muy utilizada para todo tipo de plataformas. Tiene librerías para trabajar en XR y otras para explotar todas las funcionalidades de las MQ3, mantenidas por Meta.

1.3.3. Lenguajes de programación

- Para crear scripts en Unity que permitan controlar todos los objetos del entorno, se utiliza el lenguaje C#.
- Para mostrar los resultados de una ejecución se recurre a Python, que permite crear gráficos de manera relativamente intuitiva y sencilla, en comparación con otros lenguajes.

¹www.meta.com/quest/quest-3

1.3.4. Control de versiones

Para organizar los avances en el proyecto y tener una copia de seguridad, se dispone de un repositorio de Gitbub que gestiona la actualización de ficheros.

1.4. Planificación

El proyecto, realizado desde finales de mayo a finales de noviembre de 2024, abarca todas las tareas de toma de contacto con las nuevas herramientas a disposición así como la implementación de la experiencia y el análisis del desempeño de los jugadores. En la Figura 1.3 se puede observar la evolución del TFG por fases y, en la Tabla 1.1, el desglose de los esfuerzos en las distintas partes.

Tareas		Horas
Toma de contacto	Familiarización con las MQ3	8
	Primeros pasos con Unity	17
	Total	25
Desarrollo del producto	Diseño de la experiencia	15
	Implementación de scripts	121
	Construcción de la aplicación	98
	Documentación multimedia	12
	Total	246
Análisis	Diseño de la prueba	3
	Prueba con usuarios	6
	Obtención de métricas y análisis	5
	Total	14
Elaboración de la memoria		27
TOTAL		312

Tabla 1.1: Desglose de esfuerzos en el TFG

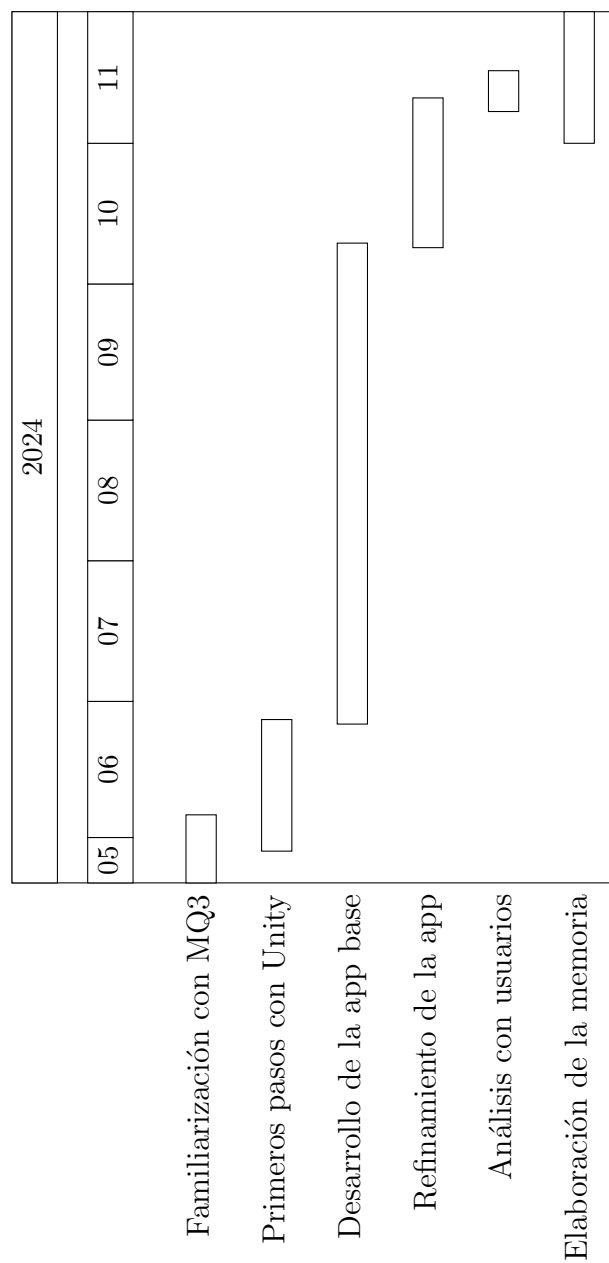


Figura 1.3: Diagrama de GANTT del TFG

Capítulo 2

Diseño e implementación

Este capítulo engloba el proceso de construcción de la aplicación, y consta de tres partes: un análisis de las herramientas utilizadas para construir la aplicación, el diseño de la experiencia para el usuario y aspectos técnicos relacionados con la implementación en Unity.

2.1. Herramientas clave

En esta sección se presentan las dos herramientas principales cuyas características resultan clave para el desarrollo de la aplicación.

2.1.1. Meta Quest 3

Las Meta Quest 3 son unas gafas de realidad mixta presentadas en 2022 y lanzadas al mercado en octubre del 2023. Las características principales de los últimos modelos de las Quest que las convierten en el HMD idóneo para un proyecto de MR son:

- En el contexto de MR, se conoce como *passthrough* a la capacidad de unas gafas de permitir visualizar el entorno real. Las MQ3 ofrecen *passthrough* a color y de alta resolución, lo que posibilita la capa de realidad en el entorno de MR. Es esta la característica que permite a las gafas insertar objetos virtuales sobre el espacio real.
- A diferencia de otras gafas, las Meta Quest 3 ofrecen una experiencia *standalone*, es decir, independiente de otros soportes o dispositivos y sin el requerimiento de conexiones físicas. Esto facilita experiencias de MR más prolongadas.
- Las MQ3 ofrecen la posibilidad de interaccionar con habitaciones. Las versiones más punteras de gafas de XR permiten escanear habitaciones y utilizarlas en las aplicaciones para personalizar la experiencia del usuario. En el caso de las Quest,

se asocia una habitación con una malla de puntos. A continuación, se completa con una serie de paredes, un suelo, un techo y distintos muebles, además de elementos de pared. Estos dos últimos van etiquetados para indicar qué son: mesa, armario, sofá, cama... para los muebles; y puertas, ventanas o misceláneo para los elementos de pared. En las MQ3 destaca un entorno amigable para la definición de habitaciones, además de una notoria persistencia de las mismas. Cada habitación va etiquetada con el nombre de un espacio común de una casa: salón, comedor, oficina, dormitorio,... En la Figura 2.1 se puede observar cómo las mesas están delimitadas por bloques que las contienen.

- Las Quest ofrecen un *hand tracking* avanzado, lo que habilita un seguimiento avanzado de las manos del usuario, permitiéndole controlar todo el sistema solo con sus manos.

En definitiva, las Meta Quest 3 son gafas idóneas para el desarrollo de experiencias inmersivas en MR, proporcionando un entorno simple y amigable, sin dependencia de dispositivos externos.

2.1.2. Unity

Unity entra en la categoría de motores de videojuegos en tiempo real. Se numeran a continuación algunas de sus características:

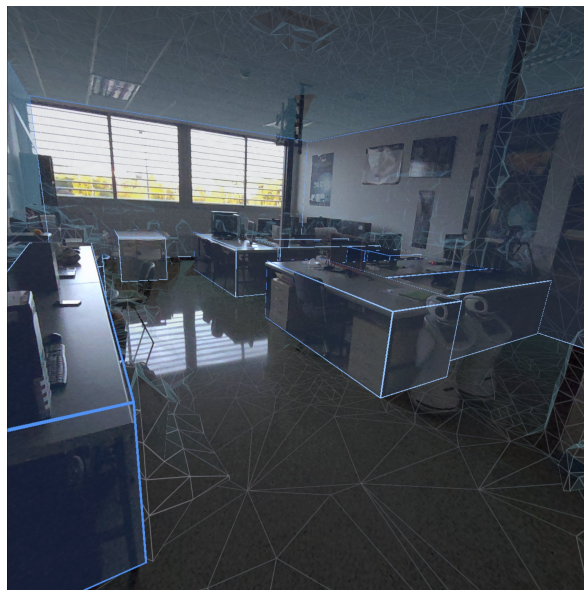


Figura 2.1: Definición en las Meta Quest 3 del Laboratorio 2.07 del Edificio Ada Byron con una malla de puntos, y su delimitación con paredes, suelo, techo y muebles, la gran mayoría etiquetados como mesas.

- Unity cuenta con varios renderizadores para entornos 2D y 3D, según la naturaleza de la aplicación. Dispone, para 3D, del renderizador URP (Universal Render Pipeline) que reduce el requerimiento de recursos para la formación de la imagen, a la vez que se mantiene una apariencia razonable en la escena.
- El entorno de Unity se basa, en grandes rasgos, en una escena y los objetos que la ocupan. Estos quedan definidos al comenzar la ejecución de la aplicación, e interactúan a lo largo del tiempo. Los objetos se actualizan en base a los *scripts* asociados a ellos.
- Estos *scripts* son, en esencia, clases: pueden tener funciones públicas para ser llamados por otros objetos del entorno. Todos los *scripts* centran su funcionamiento en dos métodos concretos: *Start()*, que es una tarea que hace el *script* al ser instanciado (es decir, al crearse el *script* o el objeto al que está asociado); y *Update()*, que es una tarea recurrente que se ejecuta a la misma velocidad que la tasa de fotogramas de la aplicación. Los *scripts* contienen atributos, que pueden ser privados o públicos, siendo estos últimos personalizables en el entorno de Unity de forma interactiva. Prácticamente todos los tipos trabajados en Unity pueden ser variables públicas, lo que facilita la tarea de implementar funcionalidades complejas en las aplicaciones. Los métodos pueden implementar funciones complejas facilitando la encapsulación de código y permitiendo paradigmas como la ejecución de corutinas (ejecución en paralelo de código respecto de otro flujo de ejecución), ya que el flujo principal de ejecución de un script funciona con un solo hilo. Hay algunos *scripts* ya predefinidos por el entorno, por ejemplo los que simulan comportamientos físicos como la gravedad. La visualización del entorno se consigue con un objeto de tipo *Camera*, desde el que se renderiza el entorno.
- Respecto a la faceta multiplataforma, Unity cuenta con el *XR Interaction Toolkit*, que sistematiza la interacción con entornos virtuales. Esto permite que las experiencias en XR también tengan portabilidad.
- Los usuarios de la comunidad publican modelos 3D para ser utilizados por otros usuarios en la *Asset Store* (web), desde donde se guardan para ser importados a continuación desde el proyecto en Unity.

Se ha de mencionar el otro motor que se ha tenido en consideración, Unreal Engine, que también tiene integración con Meta. Las razones para elegir Unity son varias: un menor consumo de recursos, una mejor compatibilidad y documentación de las APIs,

así como de la integración con MR, una adaptación multiplataforma para XR directa y una curva de aprendizaje más adecuada, sobre todo para un primer acercamiento a un motor de videojuegos.

2.1.3. Integración

Implementar una aplicación que funciona sobre XR implica manejar entradas y salidas que no aparecen en entornos como ordenadores convencionales. Trabajar sobre un dispositivo atípico requiere de herramientas que consigan explotar su potencial. Para ello, Meta ha desarrollado un SDK (Kit de Desarrollo de Software) que permite acceder a sus entradas, salidas y algunos datos de persistencia.

Este SDK es el *Meta XR All-In-One SDK*, y esta compuesto por varias interfaces que integran todas las características de las MQ3 que no figuran en la base de Unity.

- *Meta XR Core* es probablemente la API (Interfaz de Programación de Aplicaciones) más importante. Contiene el *OVRCameraRig*, un objeto que funciona como cámara y contenedor de los controles de las Quest, gestionando sus posiciones en el espacio. También controla el objeto *Passthrough*, que añade la capa de realidad a la aplicación.
- *Meta XR Audio* da acceso al audio estéreo de las Quest.
- *Meta XR Interaction* es capaz de leer las entradas de los controles. Estos pueden ser los mandos o las manos. En este proyecto, se utilizan exclusivamente los mandos como entradas del jugador al entorno.
- *Meta XR Haptics* permite mandar señales a los mandos para que vibren.
- Para la lectura e interacción con habitaciones, se utiliza otra interfaz: *Meta MR Utility Kit*, o *MRUK* por sus siglas, que encapsula la lectura y aparición en el entorno de los elementos de la habitación.

MRUK puede acceder a la habitación activa y llamar a funciones que operan en base a los límites físicos del espacio. Existen funciones para seleccionar un punto aleatorio en la habitación, incluso con un margen a superficies. También hay funciones para obtener puntos aleatorios en superficies. En este caso, se puede filtrar por la etiqueta de los elementos o por las normales de los planos para discriminar. Por ejemplo, generar puntos solo en el suelo, o en la tabla de arriba de una mesa, o en uno de los laterales de un armario. Resulta idóneo utilizar estos puntos para construir objetos en el entorno con determinada coherencia.

El componente *EffectMesh* modifica dinámicamente las superficies de la habitación activa. Puede asignarles un material o convertirlas en superficies que pueden colisionar. Puede añadir filtros para seleccionar qué partes de la habitación quiere transformar.

La documentación¹ de esta API, si bien es amplia, se encuentra a veces incompleta en lo que a explicación de los métodos respecta.

- A la hora de probar las gafas, existe *Meta XR Simulator*, un entorno integrado que imita la visión y control de las Quest para depurar la aplicación sin ser requerimiento tener las gafas a disposición.

Respecto a la construcción del proyecto, la multicompatibilidad permite exportar proyectos a distintas plataformas. En el caso de las Quest, requiere exportar un proyecto para Android. Nada más conectar las gafas al ordenador, son reconocidas y se puede construir y exportar la aplicación.

2.2. Diseño de la experiencia

Una vez descrito cómo es posible obtener experiencias inmersivas en MR haciendo uso de las APIs de Meta en Unity, se describe la experiencia que funciona sobre estos pilares y atiende las necesidades del proyecto.

2.2.1. Descripción de la experiencia

La experiencia final tiene como propósito explorar cómo distintos mecanismos de guiado de la atención (visuales, sonoros y hápticos) influyen en la capacidad de los usuarios de encontrar objetos en un entorno controlado. A su vez, este diseño busca ofrecer una herramienta que muestre de manera divulgativa conceptos sobre la atención multimodal en entornos de realidad mixta. Además, sirve para analizar las capacidades de las Meta Quest 3 en el desarrollo de experiencias XR.

El escenario de la experiencia es una habitación del mundo real, capturada y procesada mediante la API de definición de habitaciones de las Meta Quest 3. Esta API guarda sobre cada habitación sus límites (paredes, suelo y techo), así como sus muebles, (mesas, sofás, camas, armarios, monitores, puertas, ventanas, entre otros).

Sobre la escena se coloca el objeto a buscar, junto a otros objetos de relleno. La inclusión de estos últimos acompaña la tarea y evita que la localización sea trivial, dado que a día de hoy los elementos virtuales se pueden distinguir del entorno real con

¹https://developers.meta.com/horizon/reference/mruk/v71/namespace_meta_xr_mr_utility_kit

relativa facilidad. No incluirlos haría la tarea más sencilla, y por consiguiente eliminaría valor a la experiencia y a la evaluación de los mecanismos de guiado de la atención.

En cada momento, la interfaz indica al usuario cuál es el objeto que debe buscar. Una vez encontrado, el usuario debe apuntarlo con un mando de las gafas y presionar uno de los gatillos para seleccionarlo, tras lo cual se pasa a la búsqueda del siguiente objeto. La experiencia concluye una vez se han encontrado 15 objetos, mostrando una puntuación basada en el tiempo total empleado y el tiempo promedio por objeto, permitiendo así analizar el rendimiento y la influencia de los estímulos de atención implementados.

Este diseño no solo permite explorar las posibilidades técnicas de la integración entre el entorno real y los objetos virtuales, sino que también ofrece una herramienta divulgativa y accesible para públicos diversos. Se pueden apreciar algunos instantes de la experiencia en las Figuras 2.2 y 2.3.

Los mecanismos de guiado de la atención tienen como objetivo simplificar la tarea de búsqueda. Los distintos modos de la experiencia añaden estímulos en distintos canales sensoriales:

- El modo de ayuda visual añade un halo blanco al objetivo, con el fin de que destaque sobre el resto.
- El modo de ayuda sonora hace que el objeto a buscar emita un sonido.
- El modo de ayuda háptica hace que los mandos vibren si apuntan y están cerca del objetivo.

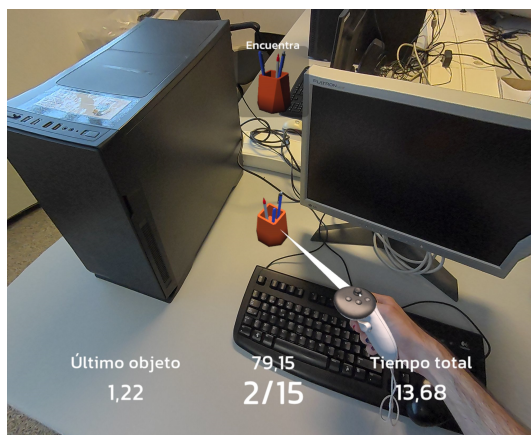


Figura 2.2: Captura de la experiencia durante la búsqueda de un objeto de mesa (cubilete con bolígrafos).

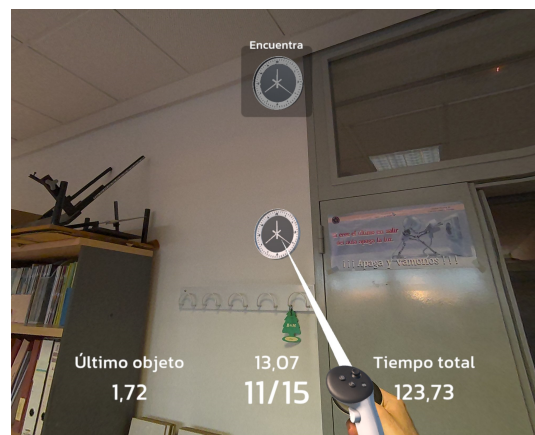


Figura 2.3: Captura de la experiencia durante la búsqueda de un objeto de pared (reloj).

2.2.2. Obtención de métricas

En paralelo a la experiencia del jugador, la aplicación recoge métricas sobre las partidas realizadas, incluyendo marcas de tiempo dónde figura la odometría de las gafas, qué objetos se buscan y dónde aparecen, además del tiempo que se emplea en encontrarlos y centrarlos en el campo de visión (si es que sucede antes de ser encontrados). Todas estas marcas se guardan en registros, cuya estructura es la siguiente:

```
started
visual on/off
sound on/off
haptic on/off
T.TTT item SCORE INDEX X.XXX Y.YYY Z.ZZZ
T.TTT odom X.XXX Y.YYY Z.ZZZ X.XXX Y.YYY Z.ZZZ
T.TTT fove T.TTT
T.TTT found T.TTT
finished
time T.TTT
avg T.TTT
```

Las líneas que empiezan por *T.TTT* son recurrentes y aparecen múltiples veces y ordenadas por el mismo valor. Los textos en mayúscula son variables numéricas, siendo aquellas con puntos de coma flotante. Se explican las distintas líneas a continuación:

- Las líneas *started* y *end* se utilizan concretamente para indicar el inicio y fin de la partida.
- Las líneas con los nombres de los distintos modos marcan con *on* si la ayuda está habilitada durante la experiencia, y *off* en caso contrario.
- La línea *item* marca cuándo aparece un objeto (*T.TTT*), cuántos lleva hasta el momento encontrados (*SCORE*), qué objeto es (cada objeto tiene un identificador asociado)(*INDEX*) y dónde está (*XYZ*).
- La línea *odom* guarda los valores de odometría en el tiempo *T.TTT*, de posición las tres primeras componentes *XYZ* y de rotación las tres últimas *XYZ*. En estos registros, la rotación se representa con una normal.
- Las líneas *fove* y *found* indican cuándo el objeto es centrado en el campo de visión (*FoV centering*) y cuando se señala, respectivamente. En ambos casos, el

primer tiempo corresponde a una marca de tiempo total. El segundo, al tiempo empleado en ejecutar esa tarea concreta.

- Las líneas *time* y *avg* muestran el tiempo tardado en encontrar los objetos y el tiempo medio por cada objeto, respectivamente.

2.2.3. Navegación

La aplicación está hecha para ser controlada por un público que no tiene por qué estar acostumbrado a entornos XR. Por ello, la interfaz y la navegación deben ser sencillas. Se puede observar el autómata de estados de la aplicación en la Figura 2.5 y el mapa de navegación en la Figura 2.4. En los siguientes puntos, se detalla brevemente cada menú. Estos menús aparecen en el mundo, concretamente en la pared más cercana al usuario en el momento que aparecen.

- La pantalla de *Inicio* permite marcar las ayudas que se van a utilizar. Permite empezar una partida o ver las reglas. Se puede ver en la Figura 2.6.
- Las *Reglas* explican brevemente la tarea a realizar. Desde aquí se puede volver a la pantalla de inicio. Se puede ver en la Figura 2.7.
- La *Partida* engloba la parte de búsqueda de objetos. Va acompañada con una interfaz que indica el tiempo transcurrido encontrando el objeto actual, el último y el total, así como los objetos que lleva encontrados. El elemento más interesante es el icono que indica cuál es el objeto a buscar.

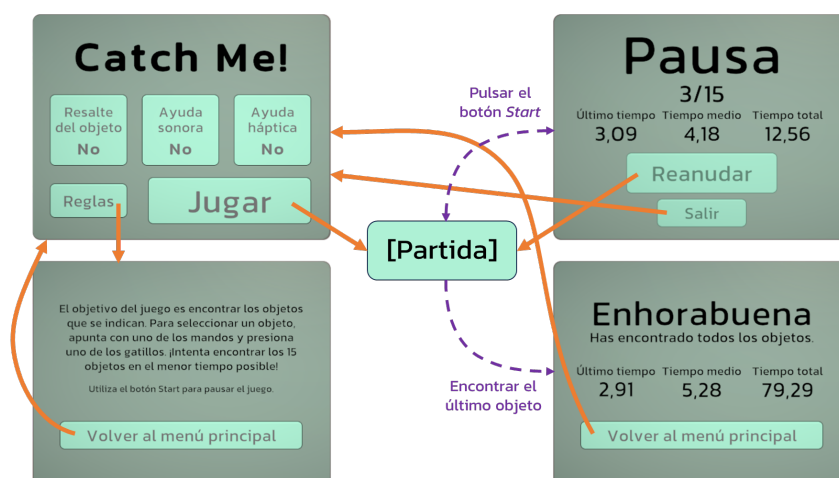


Figura 2.4: Mapa de navegación de la aplicación. Se observan las transiciones entre pantallas pulsando los distintos botones de la interfaz (y de los mandos).

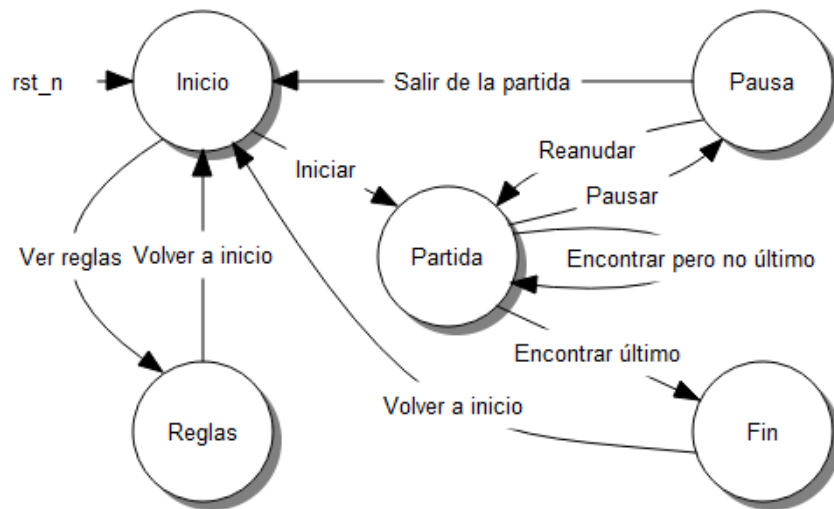


Figura 2.5: Diagrama de transición de estados de la aplicación, en forma de autómata. Cada transición va descrita con la acción que se le asocia.



Figura 2.6: Pantalla de *Inicio*. Se pueden activar ayudas o cambiar al menú de *Reglas* o empezar una *Partida*.

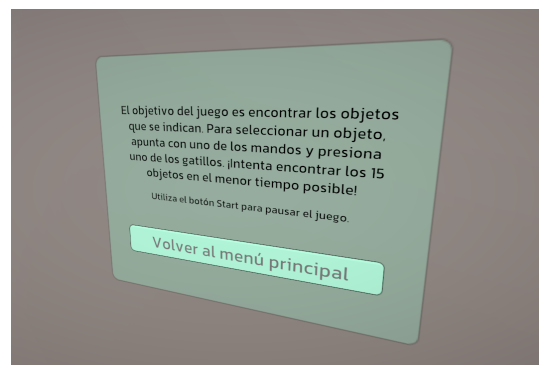


Figura 2.7: Pantalla de *Reglas*. Se observan las indicaciones de la experiencia, pudiendo volver al *Inicio*.

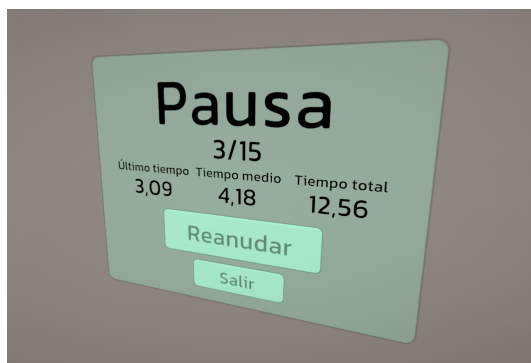


Figura 2.8: Pantalla de *Pausa*. Se pueden ver los tiempos último, medio y total, y volver a *Partida* o al *Inicio*.

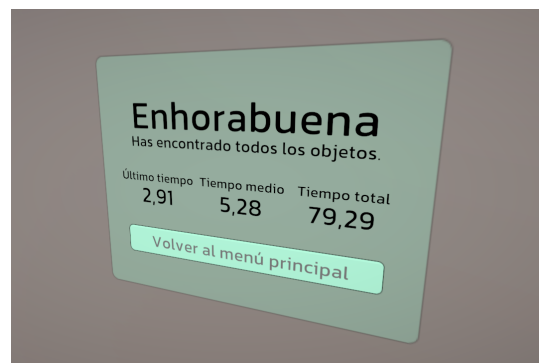


Figura 2.9: Pantalla de *Fin* de partida. Se pueden ver los tiempos último, medio y total, y volver al *Inicio*.

- El menú de *Pausa* permite consultar, al igual que durante la partida, la cantidad de objetos encontrados y los tiempos último y total, además de la media de tiempo. Permite reanudar o salir de la partida. Se puede comprobar su aspecto en la Figura 2.8.
- La pantalla de *Fin* muestra lo mismo que la de pausa, menos la cantidad de objetos encontrados. Permite volver al menú de inicio. Se puede ver en la Figura 2.9.

2.2.4. Objetos

En cada fase, se cargan entre 5 y 40 objetos, cantidad ligeramente aleatoria que va aumentando a lo largo que se van encontrando objetos en la ejecución. Los objetos que pueden aparecer son los siguientes, clasificados por dónde lo hacen:

- Objetos de mesa: taza, cubilete con bolígrafos, celo, calculadora, flexo, libreta y libros en varias posiciones.
- Objetos de pared: reloj, videocámara, panel de corcho y extintor.
- Objetos de suelo: maceta con planta y papelera.
- Objetos en el aire: avión de papel. Este objeto, en concreto, se mueve describiendo una trayectoria circular.
- Objetos de relleno: lápices, bolígrafos, subrayadores, tijeras, grapadoras y tacos de notas, de varios colores cada uno, y una goma de borrar. Hay una alta probabilidad de que aparezcan sobre una mesa, pero también es posible que aparezcan sobre otros muebles o en el suelo.

Los objetos que aparecen en la Figura 2.10 son candidatos a ser el objetivo. Los que aparecen en la Figura 2.11, sin embargo, solo aspiran a ser objetos de relleno. Se pueden ver objetos de relleno distribuidos por la habitación en la Figura 2.12.

El primer objeto en cargar será el objeto a buscar. El resto lo harán a continuación mezclando entre los demás objetos, de relleno y otros posibles candidatos, de forma que el objeto a buscar no se repita.

2.2.5. Interacción

Para interactuar en la aplicación se utilizan los mandos de las MQ3. Un rayo desde el mando cumple la función de cursor en los menús y sirve para encontrar los objetos durante la partida, en función del primer elemento que encuentre. Para hacer clic, se



Figura 2.10: Objetos candidatos a ser el objetivo. 1: taza. 2: flexo. 3: cubilete con bolígrafos. 4: calculadora. 5: portacecos. 6: reloj. 7: cámara. 8: corcho. 9: extintor. 10: avión de papel. 11: maceta con planta. 12: papelera.

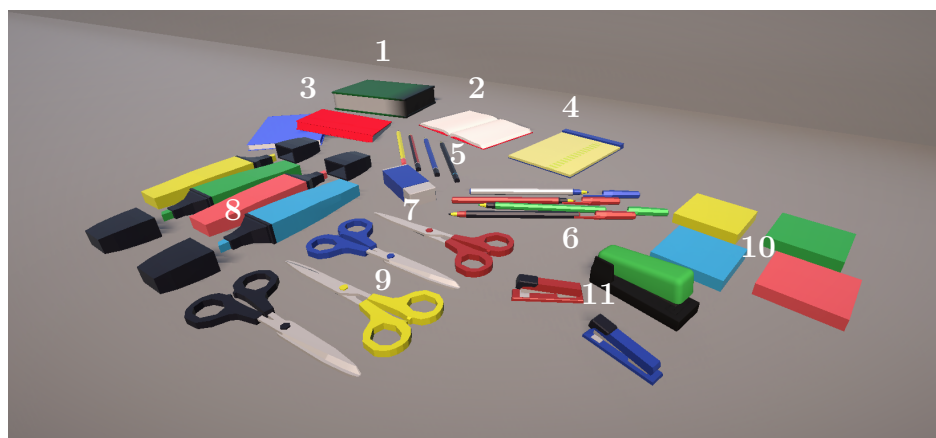


Figura 2.11: Objetos de relleno. 1: libro cerrado. 2: libro abierto. 3: par de libros. 4: libreta. 5: lápices. 6: bolígrafos. 7: goma de borrar. 8: subrayadores. 9: tijeras. 10: tacos de notas. 11: grapadoras.



Figura 2.12: Objetos de relleno repartidos por la habitación. Aparecen algunos sobre las mesas, y otros sobre el suelo. Se llena el espacio con más elementos virtuales para acomplejar la búsqueda del objetivo.

pulsa uno de los gatillos del mando que apunta. Por otro lado, para pausar y reanudar la experiencia, se presiona el botón de pausa del mando izquierdo.

Cabe recalcar que los muebles también interceptan rayos, así que si entre el mando y el objeto hay un mueble, este cortará el rayo y no llegará al objeto, lo que obliga al jugador a moverse para poder apuntar al objeto.

2.3. Implementación de la aplicación

En la sección que cierra este capítulo se profundiza sobre los detalles de implementación en Unity (con especial hincapié en las APIs de integración con las Quest) que hacen que la aplicación se ajuste al diseño previamente mencionado.

2.3.1. Lógica de la aplicación

La lógica de la aplicación queda encapsulada en un *script*, llamado *GameCore* que controla desde la gestión de pantallas hasta la generación y detección de objetos y la obtención de las trazas de usuario.

2.3.2. Estados de la aplicación

El esqueleto del *script* es el mapa de navegación, cuya máquina de estados se encuentra en la Figura 2.4. Todos los estados menos el de *Partida* son estáticos porque solo presentan menús interactivos sobre el entorno real, así que se hablará sobre ellos en conjunto.

Para cambiar entre los distintos estados, hay una función que sistematiza las transiciones. Ir del *Inicio* a la *Partida* ejecuta toda la lógica para empezar la experiencia. Volver al menú de *Inicio* desde cualquier otra pantalla hace que se borre toda la información de cualquier ejecución anterior, limpiando el entorno para iniciar la experiencia de nuevo.

Es posible salir de la *Partida* de dos formas: encontrando todos los objetos, moviéndose al estado de *Fin* y acabando con toda la lógica que hace que aparezcan objetos; o pausando la experiencia, lo que oculta los objetos y marca la aplicación como pausada, moviéndose al estado de *Pausa*. Reanudar desde el menú de *Pausa* realiza el efecto contrario: muestra los objetos, desmarca la aplicación como pausada y vuelve a la *Partida*. Para pausar y reanudar una ejecución, se ha de pulsar el botón de *Start* del mando izquierdo.

En todas las transiciones se oculta la interfaz gráfica del estado anterior y se muestra la del nuevo estado.

2.3.3. Interfaz gráfica

En Unity, un *canvas* es un lienzo en 2D presentado sobre un entorno 3D. Cada estado de la aplicación tiene un *canvas* asociado:

- Durante la *Partida*, el *canvas* está en el *Screen Space - Camera* (o coordenadas de la cámara), por lo que su posición va acorde con la de las gafas en todo momento para aparecer fijo y visible en la pantalla, cumpliendo la función de HUD (Head-Up Display).
- En el resto de estados, el *canvas* se ubica en el *World Space* (o coordenadas globales), lo que significa que se coloca en el entorno como un objeto más. Cada vez que aparece un menú sobre el mundo, lo hace sobre la pared más cercana al jugador, en paralelo a la pared.

Cada interfaz es, por lo tanto, un *canvas* al que se le asocian varios textos y botones interactivos. A cada *canvas* se le asocia también un *script* que se encarga de actualizar los valores de algunos campos, además de mostrar y ocultar componentes para la *partida* y posicionar el menú sobre una pared para el resto de estados.

El *script* para la *Partida* se llama *UIManager*. Para el resto de estados, el nombre es *MenuManager*.

2.3.4. Menús

En los distintos estados, la interfaz incluye al menos un botón para navegar entre las distintas pantallas. El *GameCore* se encarga de comprobar si, al hacer clic con un mando, el rayo choca con un botón y, en caso de ser así, ejecuta el método *OnClick()* de ese objeto. Se considera hacer clic con el mando pulsar algún botón determinado. En este caso, se utilizan los gatillos de cada mando como botón de clic.

Casi todos los botones de la interfaz se utilizan para cambiar entre estados. A estos botones se añaden tres en el menú de *Inicio* para activar y desactivar las tres ayudas.

2.3.5. Logger

Se ha diseñado un *script* llamado *Logger*, que encapsula la creación de ficheros tipo *log* o registro, y la adición de contenido a estos ficheros.

2.3.6. Partida

El estado de *Partida* es más complejo porque está compuesto por dos subestados o fases: búsqueda y descanso.

- La fase de descanso dura 1 segundo y hace la función de transición entre búsquedas. Nada más iniciar, elige aleatoriamente un objeto entre una serie de candidatos. Después, modifica la interfaz para que aparezca en el centro de la pantalla qué objeto es el siguiente a buscar, tal como aparece en la Figura 2.13. Finalmente, lanza una corutina que se encarga de preparar la escena, es decir, cargar todos los objetos que van a aparecer.



Figura 2.13: Pantalla de descanso durante la *Partida*. Se oscurece ligeramente el entorno mientras está vacío de objetos virtuales, y aparece en grande el objeto a buscar en el centro de pantalla.



Figura 2.14: Ejemplo de ayuda visual. Un halo blanco emana del objeto a buscar.

La corutina de carga de objetos se encarga de instanciar los objetos uno a uno. El primero de todos es siempre el objetivo. Si se ha habilitado la ayuda visual o sonora, se agrega al objeto un efecto de halo (pudiendo verse un ejemplo en la Figura 2.14) o un *AudioSource*, respectivamente. El resto de objetos cargan acto seguido procurando no repetir el objeto a buscar.

Cuando pasa el segundo, se despeja la interfaz, se informa al *Logger* de qué objeto se ha de buscar y donde está, se muestran los objetos en la escena y se pasa a la fase de búsqueda.

- La fase de búsqueda no tiene duración definida. Mientras la fase dura, corre y se actualiza el tiempo en la interfaz. El *Logger* recoge periódicamente la odometría de la cámara y analiza cuándo el objeto es centrado en el campo de visión (es decir, se centra en la pantalla de las Quest), si es que sucede antes de ser señalado. Todas las entradas del *Logger* van con su marca de tiempo correspondiente.

Además, si está la ayuda háptica habilitada, se actualiza la intensidad de la vibración de los mandos en función de la posición de cada uno respecto del objeto a buscar. La intensidad se calcula como $vibr = \frac{max_{dist} - dist}{max_{dist}}(\vec{m}\vec{p})$, siendo como mínimo 0 y como máximo 1. La componente fraccionaria hace referencia a la distancia del mando al objeto, incrementando conforme el mando se acerca. Si se encuentra a max_{dist} de distancia, entonces el mando no vibra. El producto escalar de vectores hace referencia al coseno del ángulo que forma el rayo que sale del mando y la recta que lo conecta con el objeto a buscar. En otras palabras, el mando vibra más conforme más apunte al objeto. La Figura 2.15 explica manifiesta este comportamiento de una forma más visual.

Una vez se encuentra el objeto, apuntándole mientras se presiona el botón de clic, se actualiza la interfaz con el nuevo tiempo último, total y la nueva puntuación. Se registra también en el *Logger* que se ha encontrado el objeto. Se destruyen los objetos y se vuelve a la fase de descanso en caso de no haber llegado al objetivo de puntos. Si se ha llegado, se va en su lugar al estado de *Fin* de partida.

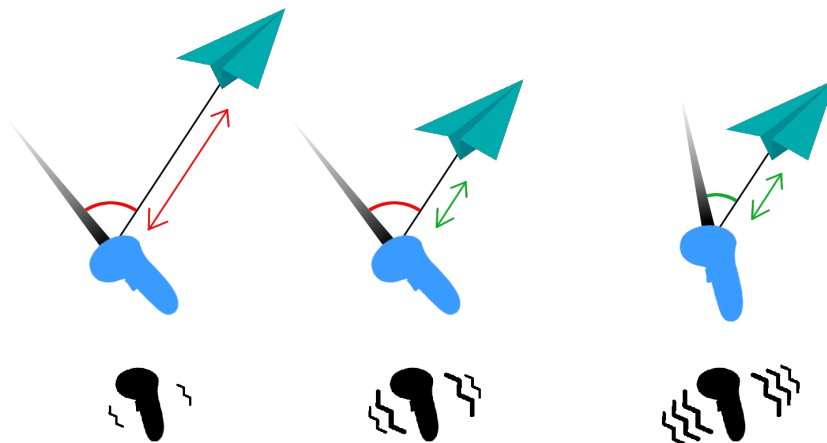


Figura 2.15: Abstracción de la intensidad de vibración del mando en la ayuda háptica. El mando apunta más conforme más se acerca al objeto y conforme mejor le apunte.

2.3.7. Objetos virtuales

A continuación se explican todas las técnicas para obtener puntos adecuados sobre los que instanciar los objetos, y que estos se coloquen correctamente:

- Para encontrar una posición en el aire, se llama a la función *MRUKRoom.GenerateRandomPositionInRoom()*, que genera un punto aleatorio en la habitación. Admite un parámetro para establecer un margen con las superficies y muebles de la habitación. Se puede ver un objeto de este tipo en la escena en la Figura 2.16.
- Para encontrar posiciones en superficies de muebles, hay que ser más específicos. Se recurre a *MRUKRoom.GenerateRandomPositionOnSurface()*, que busca un punto en las superficies de la habitación. Para discriminar entre distintas superficies hace falta utilizar sus dos parámetros que filtran por etiqueta del elemento y por la normal del plano. De aquí salen el resto de búsquedas de posiciones:
 - Para suelos, se utilizan planos tipo *FLOOR*, con la normal apuntando hacia arriba. Se puede ver un ejemplo de este tipo de objetos en la Figura 2.17.
 - Análogo para objetos sobre la mesa (tipo *TABLE*).
 - Para objetos de relleno, se buscan normales hacia arriba también, pero las superficies ocupables varían. Hay una alta probabilidad de ocupen una mesa, y otra más baja de buscar entre más superficies: mesas, camas (*BED*) y suelos.

- Las paredes buscan planos tipo *FACE_WALL* y, en su caso, normales que apuntan a laterales. Es importante guardar esta normal, ya que de ella depende que el objeto se oriente bien en la escena.

Las búsquedas en superficies no tienen en consideración los volúmenes de las habitaciones. Esto se traduce en objetos dentro de muebles, un comportamiento no deseado. Para evitarlo, se sigue la normal obtenida sobre el punto escogido para comprobar si está dentro de un mueble con la función *MRUKRoom.IsPositionInSceneVolume()*. En caso afirmativo, se descarta el punto y se busca otro hasta encontrar uno que no esté encerrado en un mueble.

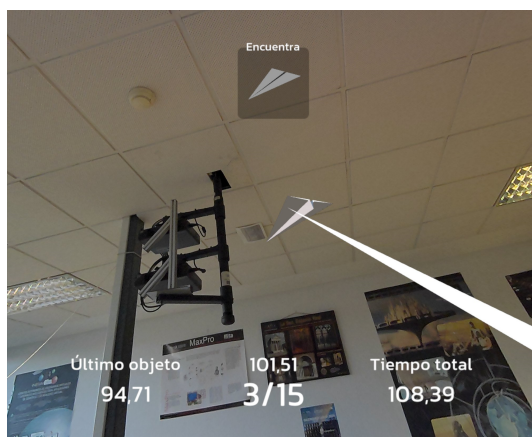


Figura 2.16: Captura de la experiencia durante la búsqueda de un objeto de aire (avión de papel).

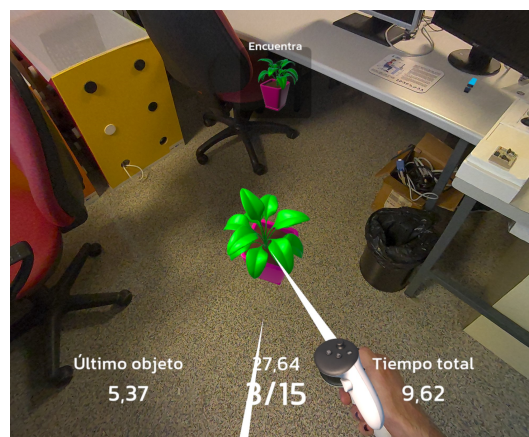


Figura 2.17: Captura de la experiencia durante la búsqueda de un objeto de suelo (maceta con planta).

Capítulo 3

Pruebas y análisis

En el siguiente capítulo, que completa el dominio de este proyecto, se prueba la aplicación con distintos sujetos. Se tomarán métricas de su desempeño en varias ejecuciones y se analizarán para cuantificar la eficacia de las distintas ayudas ofrecidas.

3.1. Pruebas a realizar

En esta sección se detalla el procedimiento a seguir para realizar un banco de pruebas sobre un conjunto de personas voluntarias.

3.1.1. Descripción de las pruebas

Las pruebas buscan comparar el rendimiento de los jugadores conforme a la aparición de distintas ayudas a la atención multimodal. Por ello, se consideran cuatro modos: *clásico*, sin ayudas; y *visual*, *sonoro* y *háptico* para el resto, cada uno activando solo la ayuda asociada a ese canal.

Cada persona realiza las ejecuciones en un orden aleatorio. Se han ordenado aleatoriamente las 24 permutaciones posibles de las ejecuciones para decidir el orden de las ejecuciones de cada participante.

Se estima que una prueba ocupa alrededor de 10 minutos, incluyendo la explicación de la experiencia, el ajuste y la toma de contacto con las gafas, las partidas en sí y las iteraciones para cambiar el modo, lo que lo hace una experiencia asequible en términos de tiempo.

Las pruebas se realizan en el laboratorio 2.07 del edificio Ada Byron. Es un laboratorio de tamaño considerable para que haya bastantes localizaciones donde colocar objetos, pero no demasiado grande para complicar la tarea de búsqueda. Además, el laboratorio cuenta con varias mesas y otros muebles. Se ha escaneado la habitación con las gafas y se han definido sobre ella todas las mesas y otros muebles

del entorno. Se pueden observar los muebles definidos sobre la habitación en la Figura 2.1.

Para cada ejecución de cada prueba se guarda únicamente la hora de realización, para a posteriori obtener el registro asociado a dicha ejecución.

3.1.2. Realización de las pruebas

Las pruebas se realizan a lo largo de varios días. Se recogen los datos de desempeño en las ejecuciones de 16 personas voluntarias, a las que se ha explicado previamente la experiencia.

3.2. Obtención de métricas

Una vez probada la aplicación con personas, se pueden recoger los registros de sus ejecuciones. Los valores de odometría de estos registros se pueden utilizar para obtener varias métricas:

- Altura media de las gafas: asociada a la de la persona que las lleva. Se obtiene promediando la coordenada Y de las gafas a lo largo de toda la ejecución.
- Rotación de las gafas: cuánto ha girado el usuario la cabeza durante la ejecución. El ángulo que giran las gafas en un intervalo concreto se consigue comparando la orientación de las gafas en el instante de tiempo inicial y el final de dicho intervalo. Para evitar pequeñas irregularidades en la medición, se comparan datos con intervalos de, al menos, una décima de segundo, por lo que se descartan la mayoría de datos.
- Rotación por unidad de tiempo: se obtiene dividiendo la rotación total por el tiempo transcurrido en realizar la ejecución.

También se obtienen otros resultados, en este caso con ayuda de una hoja de cálculo, para trabajar a la vez los datos de todas las ejecuciones: el ratio de tiempos de ejecución de todas las ejecuciones respecto de la primera, de cada ayuda respecto de sin ayudas, el tiempo promedio en encontrar cada uno de los distintos objetos y cuántas veces los objetos quedan centrados en el campo de visión antes de ser encontrados (en general y desglosado por modos).

3.3. Desarrollo de ejecuciones

Se pueden utilizar los valores de odometría y aparición de objetos de los registros para obtener el desarrollo completo de una ejecución. Se pueden ver ejecuciones de

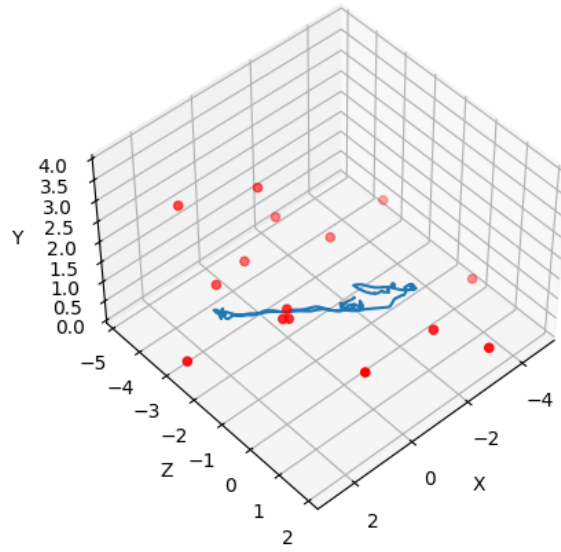


Figura 3.1: Posición de los objetos (puntos en rojo) y recorrido de las MQ3 (puntos en azul) durante la ejecución de ayuda sonora de la persona voluntaria 2. Se observa cómo la persona se ha movido por el entorno para alcanzar con los rayos objetos que no podría alcanzar quedándose en el sitio por la colisión de los rayos con muebles.

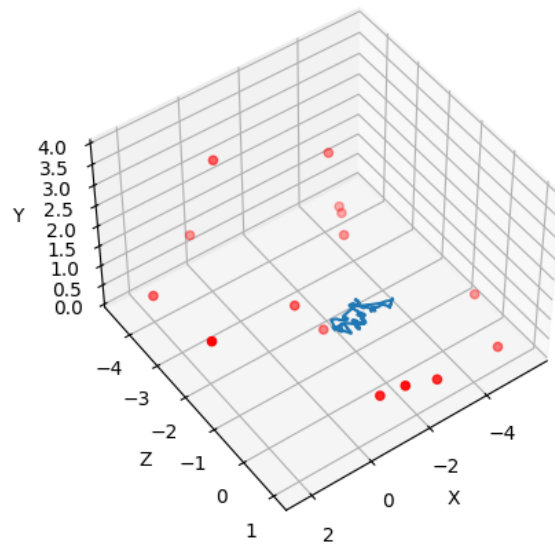


Figura 3.2: Posición de los objetos (puntos en rojo) y recorrido de las MQ3 (puntos en azul) durante la ejecución de ayuda háptica de la persona voluntaria 15. La persona ha realizado la ejecución completa desde el sitio, pues los muebles no han llegado a obstruir los rayos hacia los objetivos de esta ejecución.

partidas como la de la Figura 3.1, en la que el jugador se tiene que mover a uno de los lados de la habitación para que las mesas no intercepten el rayo hacia el objeto. En otras ejecuciones, como la de la Figura 3.2, esto no ha sido necesario.

3.4. Análisis de resultados

Los resultados son complejos y dan explicación a varios asuntos. Se pueden encontrar los valores obtenidos en el Anexo A. A continuación, se anotan las consideraciones que se han hecho promediando los datos de todas las personas voluntarias respecto a varios criterios.

3.4.1. Eficacia de las ayudas recibidas

La Figura 3.3 desglosa por modos el tiempo promedio en terminar una ejecución.

El modo clásico, sin ayudas, resulta ser el más costoso para los jugadores, aunque con una serie de datos muy variados. En general, son unos pocos datos los que incrementan en gran medida este valor.

El modo visual no reduce demasiado la media. Sin embargo, ayuda a los jugadores a centrarse en el objeto una vez entra en el campo de visión, lo cual puede explicar la menor desviación estándar en este modo.

El modo sonoro añade un estímulo conocido para el ser humano, por lo que es fácil de identificar. Este estímulo es omnidireccional, por lo que independientemente de dónde se encuentre la fuente de sonido, esta va a aportar información. Es por ello que tiene la media y la desviación estándar más bajas.

El modo háptico presenta una media similar a la del modo visual, aunque con

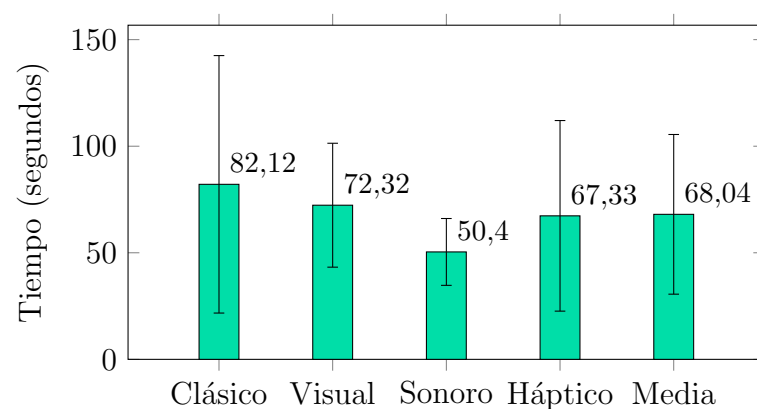


Figura 3.3: Comparación de medias y desviaciones estándar de tiempos de ejecución, agrupadas por el modo de ejecución. El modo sonoro es el mejor de media, mientras que el clásico es el peor. Además, los modos clásico y háptico presentan grandes variaciones en el tiempo.

una desviación mucho mayor. La ayuda que este modo ofrece es omnidireccional como la sonora, pero no es habitual para el ser humano como la otra. Algunas personas entienden el mecanismo rápidamente, y lo aprovechan para obtener mejores tiempos (incluso llegando a explotar su potencial, teniendo en cuenta que la vibración de cada mando es independiente). Otras personas encuentran la ayuda confusa, por lo que apenas hace alguna diferencia.

Independientemente de la ayuda ofrecida, todas reducen el tiempo de búsqueda, en mayor o menor medida.

3.4.2. Curva de aprendizaje

Si bien resulta interesante desglosar los tiempos por las ayudas recibidas, durante la prueba con usuarios se detectó que las primeras ejecuciones tienden a ser mucho más largas que las demás. Para analizar cuánto puede afectar la curva de aprendizaje a los tiempos obtenidos, se reorganizan los datos para agruparlos no por las ayudas recibidas, sino por riguroso orden de ejecución.

En la Figura 3.4 se aprecia cómo el jugador se familiariza con la mecánica después de la primera ejecución. Los jugadores tardan cada vez menos en encontrar los objetos, sin mucha disparidad entre ellos. El tiempo se reduce a un poco menos de la mitad a partir de la segunda ejecución.

La rotación de la cabeza, desglosada por modo, no da explicaciones enriquecedoras, ya que por unidad de tiempo es muy similar en todos los modos. Sin embargo, al desglosar por ejecuciones como en la Figura 3.5, se observa que en las ejecuciones que suceden a la primera la rotación por segundo es mayor.

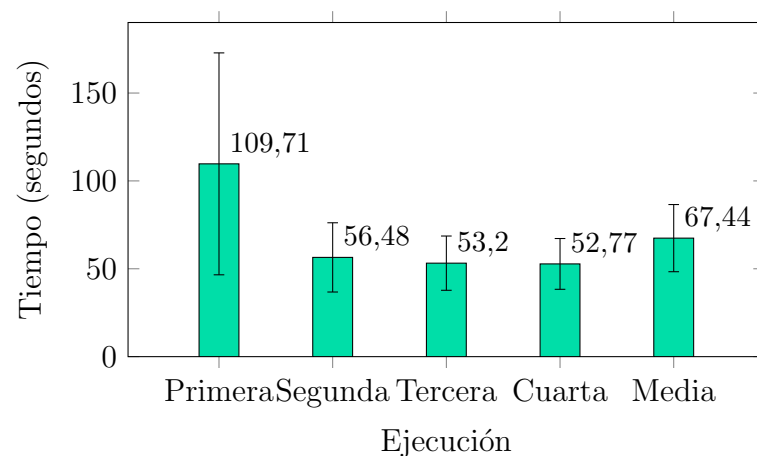


Figura 3.4: Comparación de medias y desviaciones estándar de tiempos de ejecución, agrupadas por el orden de ejecución. En general, la primera ejecución es la más dispar entre personas, aunque en general es la más costosa con diferencia. El resto de ejecuciones tardan menos de la mitad de tiempo que la primera, de media.

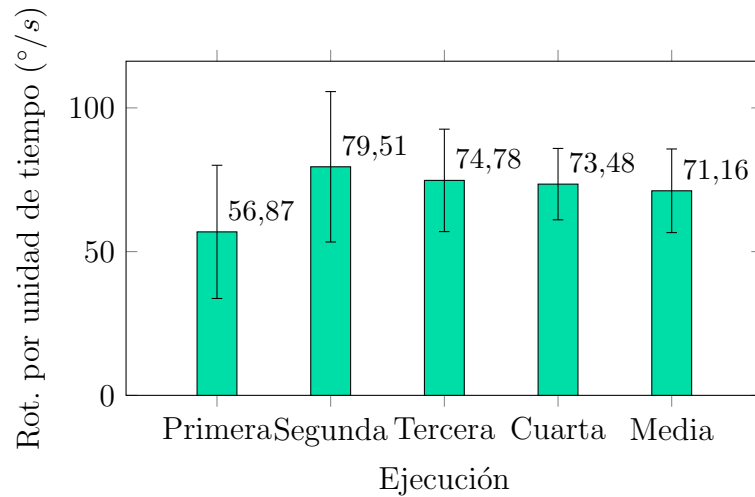


Figura 3.5: Rotación de la cabeza por unidad de tiempo, desglosada por orden de ejecución. En la primera ejecución se detecta menos rotación que en las ejecuciones que le siguen.

3.4.3. Curvas de aprendizaje según el modo

Para determinar cómo evoluciona la curva de aprendizaje para cada modo, se han promediado los tiempos para cada modo y orden de ejecución. El resultado de esta discretización se observa en la Figura 3.6.

Es notable cómo influye la curva de aprendizaje en los distintos modos de manera individual. En el caso del modo clásico, depende de muchos factores como la capacidad de cada individuo de adaptarse a los controles y al entorno, la experiencia previa con

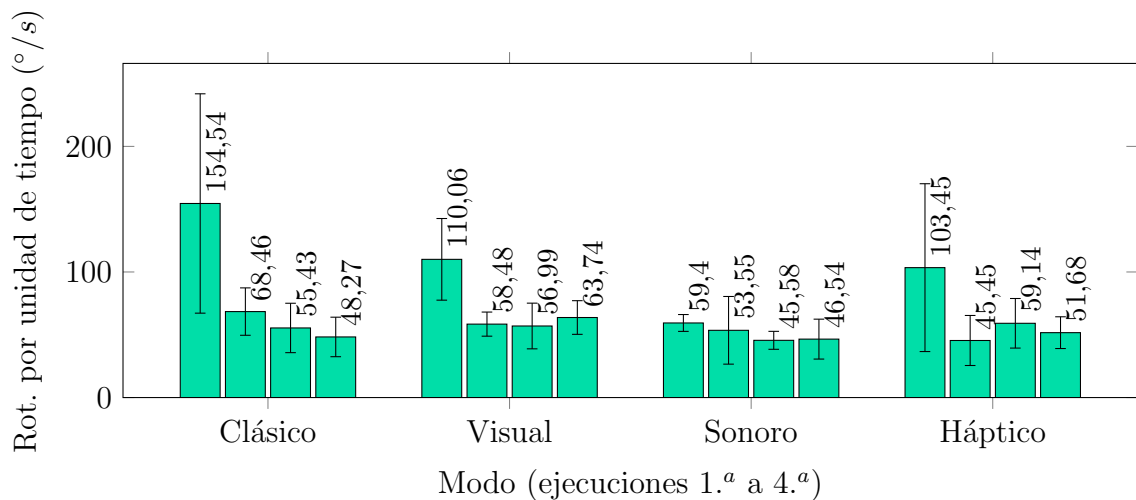


Figura 3.6: Comparación de promedios de tiempos de ejecución, desglosados por modo y orden de ejecución. La curva de aprendizaje es apreciable en todos los modos menos el sonoro. En todos los modos menos este, la primera ejecución es mucho más costosa que las demás y la variación muy alta. Las ejecuciones menos costosas también tienen poca desviación estándar.

XR, etc. Para los modos visual y háptico, depende de la rapidez con la que el usuario entienda los estímulos añadidos como ayudas para encontrar los objetos. En los tres modos mencionados el promedio y la varianza del tiempo son muy altos en la primera ejecución y decrecen drásticamente en el resto.

El modo sonoro es la excepción a la regla. Sus resultados son comparables a los del modo háptico, solo que a diferencia de este otro modo, el ser humano sí está acostumbrado a señales sonoras en el entorno real, lo que hace la labor de búsqueda más amena y simplifica la labor de aprendizaje. Todo esto se traduce en primeras ejecuciones más cortas y una varianza, en general, más pequeña.

3.4.4. Dificultad de los objetos

Son varios los objetos que aparecen a lo largo de las ejecuciones y, vistos los resultados obtenidos, no todos son igual de fáciles de encontrar. Se obtiene la media aritmética de búsqueda de cada objeto para cada modo, y se promedia entre los cuatro modos, obteniendo el tiempo medio de búsqueda por objeto.

En la Figura 3.7 se manifiestan las diferencias de tiempo entre objetos. Mientras que algunos objetos son llamativos y se encuentran rápido, otros no lo son tanto. Los motivos que pueden determinar esto son su saturación (los objetos en escala de grises se mimetizan con paredes blancas), en el caso de la calculadora y el avión su pequeño tamaño y concretamente con el avión que es un objeto móvil.

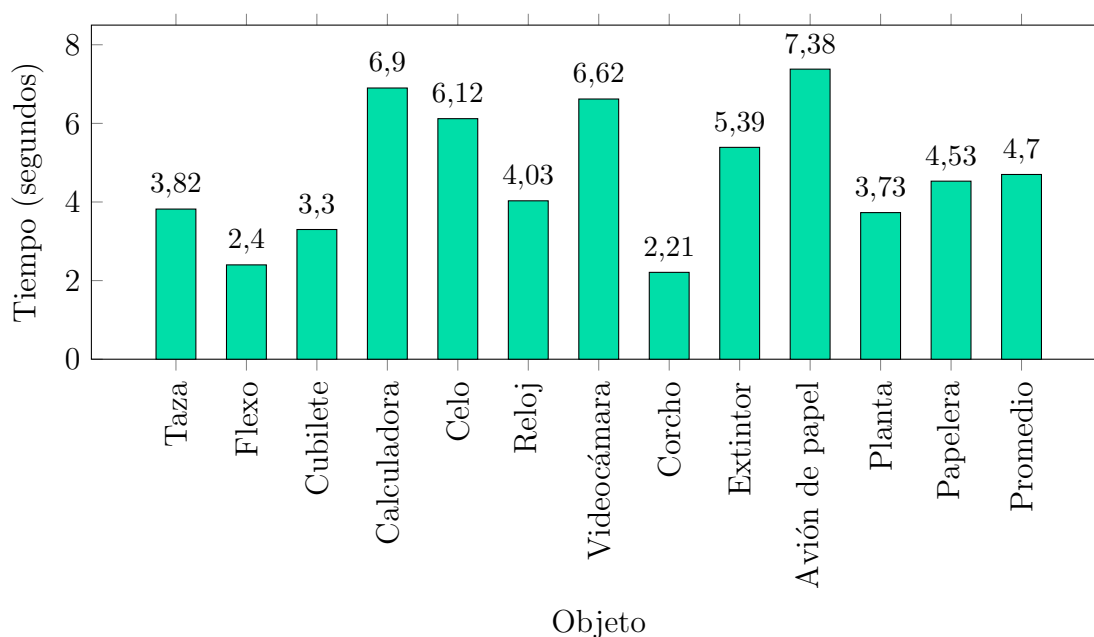


Figura 3.7: Tiempo medio de búsqueda de un objeto, desglosado por objetos. Se nota cómo con algunos, como el flexo y el corcho, se tarda poco tiempo en comparación con otros como la calculadora, la videocámara y el avión de papel.

3.4.5. Centrado del objetivo en la pantalla

Es complejo analizar el tiempo de centrado de objetos en el campo de visión, o si este es relevante. Hay que tener en cuenta que no se requiere centrar el objetivo en la pantalla para ser encontrado. En la Figura 3.8 se observa que es frecuente que en las ejecuciones algún objeto no esté centrado en ningún momento. La diferencia entre los distintos modos es mínima. Se puede atribuir un ligero descenso en los modos háptico y sonoro a una respuesta de las manos más rápida que la de la cabeza ante la presencia de estímulos multimodales y omnidireccionales.

Además, la probabilidad de centrar el objetivo en el campo de visión varía según el objeto. En la Figura 3.9 (obtenida de manera análoga a la Figura 3.7), se observa como la planta se centra mucho menos que el resto de objetos. Esto se debe a que es un objeto llamativo, grande y discretizado (solo puede aparecer en el suelo).

3.5. Opiniones de los usuarios

Tras probar la aplicación con varios usuarios, y escuchar sus opiniones, resulta de interés conocer sus aportes.

3.5.1. Potencia de vibración en el modelo háptico

Algunos usuarios consideraban que el mando vibraba al máximo cuando no estaban apuntando al objeto. La vibración es una función que depende de dos variables: la distancia en una escala lineal, y el ángulo que forman el rayo del mando con la recta que lo une con el objeto, en una escala cosenoidal.

Esto implica que apuntar perfectamente al objeto desde más lejos afecta menos que

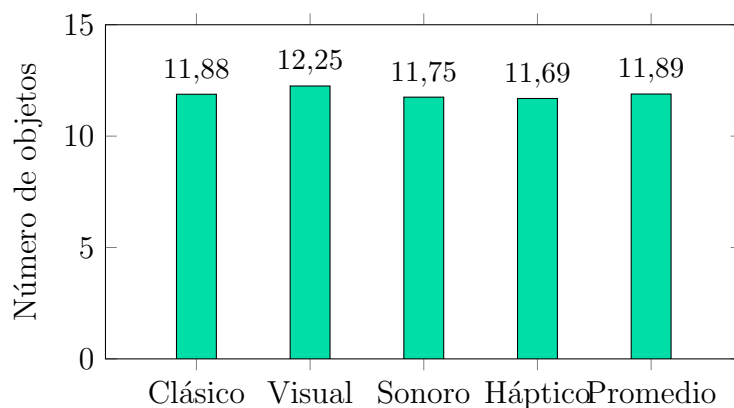


Figura 3.8: Promedio de los objetos que son centrados en la visión del usuario por partida (sobre 15 objetos), desglosado por modo. No se aprecia una elevada disparidad entre modos.

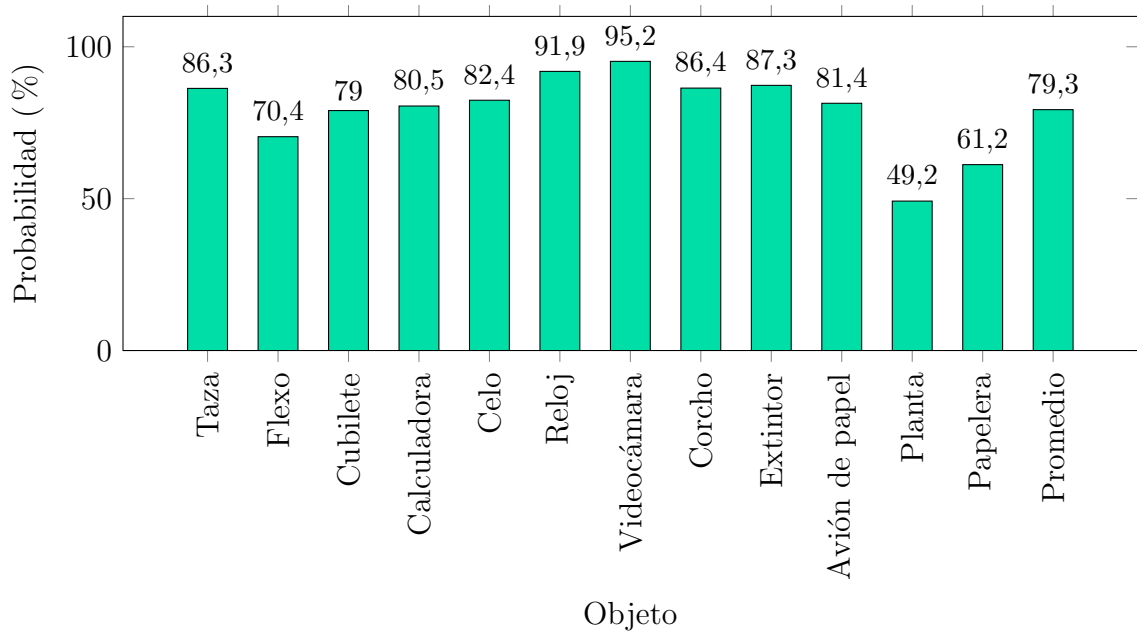


Figura 3.9: Probabilidad de centrar el objeto en la pantalla, desglosado por objetos. En concreto, la planta y la papelera no siempre se centran, al contrario que la cámara que casi siempre lo hace.

apuntar parcialmente pero de más cerca. En resumen, se *castiga* poco no apuntar bien al objeto, por lo que sería conveniente ajustar la función en base al ángulo, como un coseno cuadrado o una función gaussiana.

3.5.2. Muebles sin oclusión y como interceptores de rayos

Otro tema a destacar es el hecho de que los objetos virtuales se muestran por delante de otros objetos reales, cuando teóricamente se encuentran detrás. Esto era palpable en las pruebas de la aplicación puesto que hay muchas mesas en la habitación, pero en ningún caso se ocultan los objetos durante la experiencia.

Por otro lado, las mesas (y resto de muebles) definidos en la habitación interceptan los rayos, impidiendo apuntar a objetos que se encuentren detrás de los muebles. Esto obliga al jugador a recolocarse para seleccionar el objeto adecuadamente.

Aunque se podría haber solucionado el problema de visión con una textura de oclusión de objetos traseros, este problema va ligado a una limitación de la definición de habitaciones de las Quest, en la que todos los muebles son prismas rectangulares. La gran mayoría de mesas tienen espacio libre por debajo de esta (pueden ser mesas con patas, escritorios con espacio para reposar las piernas, etc.), lo que teóricamente significa espacio libre para colocar objetos. Sin embargo, por la simplificación de MRUK toda la mesa queda ocupada en la simulación. Esto impide colocar objetos en este espacio. Por la misma regla, aplicar una textura de oclusión sobre elementos que

no están adecuados a la realidad puede dar lugar al cubrimiento de objetos que en condiciones generales serían visibles (como utilizar una mesa con patas como oclisor de visión cuando la parte de abajo de la mesa no ocluye la vista, a excepción de las patas).

Capítulo 4

Conclusiones

Este Trabajo de Fin de Grado (TFG) ha consistido en la confección de una experiencia en MR que permite conocer y medir la atención multimodal en base a tareas de búsqueda de objetos, así como el análisis de la eficacia de distintos mecanismos que procuran dirigir la atención.

La mayor parte del proyecto ha consistido en la creación de la aplicación, en Unity, para las Meta Quest 3. Tanto Unity como la XR son temas que no se conocían en profundidad al inicio del TFG, por lo que fue necesario invertir esfuerzos en comprender cómo funciona el entorno.

Se ha trabajado con un SDK reciente y compatible con las Quest 3 para integrar las características de las gafas de XR con la aplicación en Unity, desde la detección de los mandos hasta la gestión de la habitación para habilitar experiencias de MR. Comprender la carga de la habitación y el posterior uso de las funciones para trabajar sobre ella fue otro punto de inflexión en el desarrollo del TFG.

Para crear la aplicación, se ha construido un entorno sólido para trabajar con MR. Este entorno se *moldea* para convertirlo en la experiencia deseada. Se añaden los estados de la aplicación, los objetos, las distintas ayudas y se prueba el conjunto hasta obtener la experiencia final: una aplicación de inmersión en MR basada en tareas de búsqueda de objetos, con la posibilidad de añadir estímulos multimodales (visuales, sonoros o hápticos) para dirigir la atención del usuario hacia el objeto.

La aplicación obtiene datos de las ejecuciones realizadas. El último paso del TFG consiste en la recopilación de las trazas de ejecución, probando la experiencia con un grupo de personas voluntarias, y su posterior análisis.

A grandes rasgos, se observa que la ayuda sonora es la más útil por dos motivos: es intuitiva y conocida, lo que hace que los jugadores la comprendan con más facilidad. La ayuda háptica no es tan habitual e intuitiva como la sonora, lo que hace que no dé tan buenos resultados, aunque es la segunda mejor ayuda. Ambas ayudas funcionan mejor que la ayuda visual, que a diferencia de las otras dos ayudas, no es omnidireccional.

Además, la curva de aprendizaje juega un papel clave para entender el comportamiento de los usuarios. Por lo general, los jugadores tardan mucho tiempo en hacer la primera ejecución y mucho menos en las demás.

Por otro lado, los objetos más llamativos tienden a tener menores tiempos de búsqueda. Algunos de los objetos son tan llamativos que se encuentran sin siquiera llegar a focalizar el campo de visión en ellos.

En conclusión, se ha creado una experiencia en MR de búsqueda espacial con mecanismos cuya capacidad de dirigir la atención ha sido probada.

4.1. Limitaciones y trabajo futuro

La mayor parte de los problemas a la hora de desarrollar la aplicación se han dado con el SDK de Meta para integrar las gafas con Unity. Este SDK es novedoso y se encuentra en constante actualización (durante la realización del TFG, ha pasado de la versión 66.0 a la 71.0).

Concretamente, el *Meta MR Utility Kit* es un SDK muy reciente, habiendo sido lanzado en diciembre del 2023. Se adecúa a la personalización que ofrecen las gafas, que es suficiente para definir habitaciones con muebles muy sencillos. La definición de muebles, tal y como se comentaba en el Subapartado 3.4.2, está limitada a prismas rectangulares para volúmenes y rectángulos para elementos de pared. La abstracción hace que, en muebles como una mesa de cuatro patas, el espacio entre las patas se considere ocupado. En un entorno ideal, esto no sería así.

Querer cambiar esta característica implica acomplejar el proceso de definición de habitaciones de las Quest al usuario promedio. Podrían ofrecerse una serie de modelos de mesa adecuados a esquemas básicos como mesas con patas, escritorios con espacio libre, o una cajonera, etc. Esto implica que la habitación tendría volúmenes más finos en vez de un bloque enorme para definir el mueble, lo que requiere analizar más el entorno y replantear algunas funciones de la API. Por todo lo mencionado previamente, será una tarea difícil pero reconfortante para la calidad de ciertas aplicaciones una gestión más detallada de los muebles de la habitación.

En el momento que los muebles se puedan definir con mejor detalle, se puede volver al planteamiento del Subapartado 3.4.2, en el cual se sugiere que los muebles deberían impedir la visión de los objetos que tienen detrás.

En la misma línea está la gestión de objetos virtuales sobre objetos reales. Es una tarea difícil de gestionar porque implica detectar los objetos reales sobre las superficies para determinar dónde no pueden aparecer objetos virtuales. El problema agrava cuando se generan objetos en muebles que no está viendo la cámara en el momento.

Posibles soluciones podrían ser que se guarde constancia de la presencia de objetos o se haga un preprocesado conforme aparecen los objetos en el campo de visión.

Respecto a las señales multimodales, convendría ahondar más en algunas de ellas. La señal visual se ha tratado como un resalte del objeto y no como un caso de iluminación global, fruto del uso del URP para renderizar, lo que dificulta la aplicación de un sistema de iluminación global que se comporte de manera adecuada con las gafas. Más realista sería una posible iluminación global si se detallan mejor los muebles.

La señal háptica ha servido de maneras inusuales a los jugadores: algunos no la entendían mientras que otros la aprovechaban más de la cuenta. Resultaría interesante entender la sensibilidad de la piel humana a la vibración para comprender en qué nivel se nota la vibración (con un enfoque similar al de los sonidos) para adecuar la intensidad de vibración. En un marco más sencillo, quizá se puedan probar otras funciones para hacer la vibración más precisa respecto de la orientación.

En definitiva, una aplicación construida sobre un SDK tiene mucho margen de mejora en base a las nuevas funcionalidades que el SDK pueda recibir. Si bien los estímulos ya probados ayudan a los usuarios, se puede profundizar más en alguno de ellos, incluso plantear alguno nuevo.

En general, los resultados muestran tendencias claras que sugieren cómo los diferentes mecanismos de guiado de atención (visuales, sonoros y hápticos) pueden influir en la localización de objetos en entornos XR. Si bien estos resultados son los deseados, es importante destacar que el objetivo principal de este TFG no es profundizar en la eficacia de estos mecanismos, sino desarrollar una experiencia interactiva que permita explorar y demostrar el uso de tecnologías XR en este contexto.

Dado el carácter exploratorio del trabajo y el reducido tamaño de la muestra, no se ha realizado un análisis estadístico formal. Aunque estas observaciones son prometedoras, para obtener conclusiones más sólidas sería necesario realizar un estudio con una muestra más amplia y análisis estadísticos detallados. A pesar de ello, este proyecto sienta una base inicial útil para futuras investigaciones y aplicaciones en el diseño de experiencias XR relacionadas con la atención multimodal.

Capítulo 5

Bibliografía

- [1] Daniel Martin, Sandra Malpica, Diego Gutierrez, Belen Masia, and Ana Serrano. Multimodality in vr: A survey. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–36, 2022.
- [2] Edurne Bernal-Berdun, Daniel Martin, Sandra Malpica, Pedro J. Perez, Diego Gutierrez, Belen Masia, and Ana Serrano. D-sav360: A dataset of gaze scanpaths on 360° ambisonic videos. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–11, 2023.
- [3] Edurne Bernal-Berdun, Jorge Pina, Mateo Vallejo, Ana Serrano, Daniel Martin, and Belen Masia. Avisal360: Audiovisual saliency prediction for 360° video. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–10, 2024.

Lista de Figuras

1.1. Componentes de las Meta Quest 3	1
1.2. Ejemplo de <i>scanpath</i> y mapa de saliencia	3
1.3. Diagrama de GANTT del TFG	6
2.1. Definición del Laboratorio 2.07 del Ada Byron en las Meta Quest 3 . .	8
2.2. Experiencia durante la búsqueda de un objeto de mesa	12
2.3. Experiencia durante la búsqueda de un objeto de pared	12
2.4. Mapa de navegación	14
2.5. Diagrama de transición de estados de la aplicación	15
2.6. Pantalla de <i>Inicio</i>	15
2.7. Pantalla de <i>Reglas</i>	15
2.8. Pantalla de <i>Pausa</i>	15
2.9. Pantalla de <i>Fin</i> de partida	15
2.10. Objetos candidatos a ser el objetivo	17
2.11. Objetos de relleno	17
2.12. Distribución de objetos de relleno por la escena	18
2.13. Pantalla de descanso durante la pantalla de <i>Partida</i>	20
2.14. Ejemplo de ayuda visual sobre un objeto	21
2.15. Abstracción de la intensidad de vibración del mando en la ayuda háptica	22
2.16. Experiencia durante la búsqueda de un objeto de aire	24
2.17. Experiencia durante la búsqueda de un objeto de suelo	24
3.1. Representación de objetos y movimientos del jugador durante la ejecución de ayuda sonora de la personas voluntaria 2	27
3.2. Representación de objetos y movimientos del jugador durante la ejecución de ayuda háptica de la personas voluntaria 15	27
3.3. Comparación de tiempos de ejecución, agrupados por modo	28
3.4. Comparación de tiempos de ejecución, agrupados por orden de ejecución	29
3.5. Comparación de rotaciones de la cabeza, desglosadas por orden de ejecución	30

3.6. Comparación de tiempos de ejecución, desglosados por modo y orden de ejecución	30
3.7. Tiempo medio de búsqueda por objetos	31
3.8. Promedio de objetos que son centrados en la visión del usuario por partida, desglosado por modo	32
3.9. Probabilidad de centrar el objeto en la pantalla, desglosado por objetos	33

Lista de Tablas

1.1. Desglose de esfuerzos en el TFG	5
A.1. Resultados de tiempo de las ejecuciones de los usuarios, desglosados por modo	48
A.2. Resultados de altura y rotación de la cabeza de las ejecuciones de los usuarios, desglosados por modo	49
A.3. Resultados de tiempo de las ejecuciones de los usuarios, desglosados por orden de ejecución	50
A.4. Resultados de rotación de la cabeza de las ejecuciones de los usuarios, desglosados por orden de ejecución	51
A.5. Resultados de entrada al campo de visión y búsqueda de los objetos desglosados por objeto y modo	52

Anexos

Anexos A

Métricas de la prueba con usuarios

Se ha realizado la prueba con un total de 16 usuarios. Aunque se muestran datos sobre las ejecuciones individuales así como el orden de ejecución, se mantiene el anonimato de las personas voluntarias.

A continuación se muestran todos los valores brutos recogidos de su desempeño, desglosados en varias categorías: agrupados por modo (en las Tablas A.1 y A.2) o por orden de ejecución (en las Tablas A.3 y A.4).

Además, se muestran los tiempos de centrado en el campo de visión (FoVc) y de búsqueda del objeto, además de la probabilidad de centrado en el campo de visión, desglosado tanto por objeto como por modo, en la Tabla A.5.

Test #	Orden de ejecución	Tiempo de ejecución (s)				Ratio de tiempo respecto del clásico				
		Clásico	Visual	Sonoro	Háptico	Prom.	Visual	Sonoro	Háptico	Med. arm.
1	C→S→H→V	130,74	71,98	88,77	45,18	84,17	0,68	0,35	0,55	0,53
2	V→H→C→S	85,84	98,59	51,95	71,54	76,98	0,73	0,87	0,53	0,71
3	C→H→V→S	101,27	84,08	66,38	47,97	74,93	0,47	0,83	0,66	0,65
4	H→C→S→V	42,72	51,99	47,77	72,02	53,63	0,59	0,66	0,72	0,66
5	C→V→S→H	283,92	70,53	37,13	73,25	116,21	0,25	0,13	0,26	0,21
6	S→V→C→H	45,07	61,90	64,25	41,80	53,26	0,96	0,70	0,65	0,77
7	V→H→S→C	32,94	101,53	55,24	24,30	53,50	0,24	0,54	0,32	0,37
8	V→S→C→H	47,58	157,24	42,36	50,74	74,48	0,27	0,30	0,32	0,30
9	S→H→C→V	35,55	78,27	51,75	37,97	50,89	0,73	0,69	1,51	0,98
10	H→V→C→S	63,10	50,75	36,67	39,71	47,56	1,28	1,59	0,92	1,26
11	S→C→H→V	66,98	52,70	62,21	73,09	63,75	1,08	1,17	0,85	1,03
12	V→C→S→H	86,13	82,86	40,05	43,34	63,10	1,04	0,48	0,52	0,68
13	C→S→V→H	102,22	45,39	25,37	49,28	55,57	0,25	0,44	0,48	0,39
14	H→S→V→C	64,38	50,65	57,68	136,31	77,26	0,42	0,37	0,47	0,42
15	H→V→S→C	47,49	50,75	47,70	65,75	52,92	0,77	0,73	0,72	0,74
16	H→C→V→S	78,01	47,85	31,14	204,98	90,50	0,38	0,23	0,15	0,26
Media		82,12	72,32	50,40	67,33	68,04	0,47	0,44	0,45	0,45
Desviación típica		60,40	29,07	15,68	44,71	37,46	-	-	-	-

Tabla A.1: Resultados de tiempo de las ejecuciones de los usuarios, desglosados por modo

Test	Orden	Altura prom.	Rotación de la cabeza (°)				Rotación por ud. de tiempo (°/s)					
			Clásico	Visual	Sonoro	Háptico	Prom.	Clásico	Visual	Sonoro	Háptico	Prom.
1	C→S→H→V	1,43	5600	4565	4698	2453	4329	42,83	63,42	52,92	54,29	53,37
2	V→H→C→S	1,65	5199	6098	3341	3930	4642	60,57	61,85	64,31	54,93	60,42
3	C→H→V→S	1,57	3907	5385	4441	2549	4071	38,58	64,05	66,90	53,14	55,67
4	H→C→S→V	1,63	4077	5329	4989	6103	5125	95,44	102,50	104,44	84,74	96,78
5	C→V→S→H	1,43	17465	6005	3416	4688	7894	61,51	85,14	92,00	64,00	75,66
6	S→V→C→H	1,63	3261	4914	3799	3292	3817	72,35	79,39	59,13	78,76	72,41
7	V→H→S→C	1,65	2486	5236	3426	1581	3182	75,47	51,57	62,02	65,06	63,53
8	V→S→C→H	1,65	4033	11840	4350	4534	6189	84,76	75,30	102,69	89,36	88,03
9	S→H→C→V	1,53	3219	6086	4238	3255	4200	90,55	77,76	81,89	85,73	83,98
10	H→V→C→S	1,54	4402	3235	2497	2503	3159	69,76	63,74	68,09	63,03	66,16
11	S→C→H→V	1,48	4243	3643	4319	4987	4298	63,35	69,13	69,43	68,23	67,53
12	V→C→S→H	1,62	6616	5807	2366	2603	4348	76,81	70,08	59,08	60,06	66,51
13	C→S→V→H	1,63	6650	4087	3940	3720	4599	65,06	90,04	155,30	75,49	96,47
14	H→S→V→C	1,59	4842	4400	4556	9286	5771	75,21	86,87	78,99	68,12	77,30
15	H→V→S→C	1,65	2862	3486	2506	3764	3155	60,27	68,69	52,54	57,25	59,68
16	H→C→V→S	1,58	3835	1869	1581	7790	3769	49,16	39,06	50,77	38,00	44,25
Media		1,57	5474	5556	3832	3554	4604	69,00	73,38	79,86	68,99	72,81
Desviación típica		0,08	3493	2150	983	2059	1241	15,89	15,45	26,98	13,86	15,15

Tabla A.2: Resultados de altura y rotación de la cabeza de las ejecuciones de los usuarios, desglosados por modo

Test #	Orden de ejecución	Tiempo de ejecución (s)					Ratio de tiempo respecto 1. ^a ej				
		1. ^a	2. ^a	3. ^a	4. ^a	Prom.	2. ^a	3. ^a	4. ^a	Med. arm.	
1	C→S→H→V	130,74	88,77	45,18	71,98	84,17	0,55	0,68	0,35	0,53	
2	V→H→C→S	98,59	71,54	85,84	51,95	76,98	1,15	0,61	0,83	0,86	
3	C→H→V→S	101,27	47,97	84,08	66,38	74,93	0,83	0,66	0,47	0,65	
4	H→C→S→V	72,02	42,72	47,77	51,99	53,63	1,22	1,12	1,69	1,34	
5	C→V→S→H	283,92	70,53	37,13	73,25	116,21	0,25	0,13	0,26	0,21	
6	S→V→C→H	64,25	61,90	45,07	41,80	53,26	1,37	1,43	0,93	1,24	
7	V→H→S→C	101,53	24,30	55,24	32,94	53,50	3,08	1,68	0,74	1,83	
8	V→S→C→H	157,24	42,36	47,58	50,74	74,48	3,30	0,89	1,07	1,75	
9	S→H→C→V	51,75	37,97	35,55	78,27	50,89	2,20	1,46	1,07	1,58	
10	H→V→C→S	39,71	50,75	63,10	36,67	47,56	0,80	0,58	0,63	0,67	
11	S→C→H→V	62,21	66,98	73,09	52,70	63,75	0,79	0,93	1,09	0,94	
12	V→C→S→H	82,86	86,13	40,05	43,34	63,10	0,96	0,46	0,50	0,64	
13	C→S→V→H	102,22	25,37	45,39	49,28	55,57	0,44	0,25	0,48	0,39	
14	H→S→V→C	136,31	57,68	50,65	64,38	77,26	0,79	0,90	2,12	1,27	
15	H→V→S→C	65,75	50,75	47,70	47,49	52,92	1,07	1,00	1,38	1,15	
16	H→C→V→S	204,98	78,01	47,85	31,14	90,50	0,61	0,40	2,63	1,21	
	Media	109,71	56,48	53,20	52,77	68,04	0,79	0,54	0,69	0,74	
	Desviación típica	63,14	19,70	15,44	14,45	18,47	-	-	-	-	

Tabla A.3: Resultados de tiempo de las ejecuciones de los usuarios, desglosados por orden de ejecución

Test	Orden	Rotación de la cabeza (°)					Rotación por ud. de tiempo (°/s)				
		1. ^a	2. ^a	3. ^a	4. ^a	Prom.	1. ^a	2. ^a	3. ^a	4. ^a	Prom.
1	C→S→H→V	5600	4698	2453	4565	4329	42,83	52,92	54,29	63,42	53,37
2	V→H→C→S	5199	3930	5199	3341	4417	52,73	54,93	60,57	64,31	58,14
3	C→H→V→S	3907	2549	5385	4441	4071	38,58	53,14	64,05	66,90	55,67
4	H→C→S→V	4077	4077	4989	5329	4381	56,61	95,44	104,44	102,50	89,75
5	C→V→S→H	17465	6005	3416	4688	7894	61,51	85,14	92,00	64,00	75,66
6	S→V→C→H	3261	4914	3261	3292	3812	50,75	79,39	72,35	78,76	70,31
7	V→H→S→C	2486	1581	3426	2486	2495	24,49	65,06	62,02	75,47	56,76
8	V→S→C→H	4033	4350	4033	4534	4238	25,65	102,69	84,76	89,36	75,61
9	S→H→C→V	3219	3255	3219	6086	3945	62,20	85,73	90,55	77,76	79,06
10	H→V→C→S	4402	3235	4402	2497	4013	110,85	63,74	69,76	68,09	78,11
11	S→C→H→V	4243	4243	4987	3643	4279	68,20	63,35	68,23	69,13	67,23
12	V→C→S→H	6616	6616	2366	2603	4550	79,85	76,81	59,08	60,06	68,95
13	C→S→V→H	6650	3940	4087	3720	4599	65,06	155,30	90,04	75,49	96,47
14	H→S→V→C	4842	4556	4400	4842	4599	35,52	78,99	86,87	75,21	69,15
15	H→V→S→C	2862	3486	2506	2862	2951	43,53	68,69	52,54	60,27	56,26
16	H→C→V→S	3835	3835	1869	1581	3180	18,71	49,16	39,06	50,77	39,42
Media		5474	4107	3940	3940	4386	56,87	79,51	74,78	73,48	71,16
Desviación típica		3493	1210	1105	1215	1153	23,17	26,16	17,82	12,42	14,55

Tabla A.4: Resultados de rotación de la cabeza de las ejecuciones de los usuarios, desglosados por orden de ejecución

Modo	Métrica	Objeto												Promedio
		1	2	3	4	5	6	7	8	9	10	11	12	
		Taza	Flexo	Cubil.	Calc.	Celo	Reloj	Cám.	Corc.	Ext.	Avión	Pla.	Papel.	
Clás.	% FoVc	89	81	71	95	74	87	100	82	89	95	35	58	80
	Tie. FoVc	2,22	1,45	2,14	5,20	2,51	1,80	4,12	0,96	2,13	1,52	1,93	1,97	2,33
	Tie. búsq.	3,14	1,94	2,73	9,52	5,09	3,26	10,13	2,06	6,75	12,40	4,04	5,82	5,57
Vis.	% FoVc	89	73	96	66	89	100	90	96	79	86	53	58	81
	Tie. FoVc	1,88	2,52	2,60	2,82	3,28	3,48	3,30	2,14	2,66	1,64	2,47	1,32	2,51
	Tie. búsq.	3,76	2,77	4,24	5,20	5,33	6,88	7,48	3,02	6,01	7,12	3,45	4,08	4,94
Son.	% FoVc	89	71	80	75	75	86	100	81	87	80	50	67	78
	Tie. FoVc	2,01	1,60	1,30	2,12	2,45	1,32	1,98	1,48	1,02	2,01	0,83	1,94	1,67
	Tie. búsq.	3,26	2,33	3,47	4,40	3,52	2,60	5,10	1,80	3,26	3,82	3,34	4,07	3,41
Hápt.	% FoVc	78	56	69	87	92	94	91	86	95	65	59	63	78
	Tie. FoVc	2,01	1,62	1,60	2,76	7,72	2,33	2,76	1,17	3,05	1,43	2,27	2,14	2,57
	Tie. búsq.	5,11	2,56	2,75	8,49	10,55	3,39	3,80	1,96	5,52	6,19	4,10	4,15	4,88
Prom.	% FoVc	86	70	79	81	82	92	95	86	87	81	49	61	79
	Tie. FoVc	2,03	1,80	1,91	3,23	3,99	2,23	3,04	1,44	2,22	1,65	1,87	1,85	2,27
	Tie. búsq.	3,82	2,40	3,30	6,90	6,12	4,03	6,62	2,21	5,39	7,38	3,73	4,53	4,70

Tabla A.5: Resultados de entrada al campo de visión y búsqueda de los objetos desglosados por objeto y modo