E.T.S. de Ingeniería Industrial,

Informática y de Telecomunicación

Métodos de percepción de la profundidad en realidad aumentada.

Grado en Ingeniería Informática

Trabajo Fin de Grado

Autor: Xabier Jiménez Cuesta

Directoras: Iñigo Ezcurdia, Sebastián Marichal

Pamplona, 7 de Septiembre 2021

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Contenido

AGRADECIMIENTOS 5
1. INTRODUCCIÓN 6
1.1 RESUMEN 6
1.2 PALABRAS CLAVE 6
1.3 OBJETIVO 6
1.4 HISTORIA DE LA REALIDAD AUMENTADA 6
1.4.1 DEFINICIÓN 6
1.4.2 ORÍGENES 7
1.4.3 ESTADO ACTUAL 8
1.4.4 DESARROLLO Y USOS FUTUROS 8
2. PERCEPCIÓN DE LA PROFUNDIDAD 10
2.1 SEÑALES OCULOMOTORAS 10
2.1.1 ACOMODACIÓN 10
2.1.2 CONVERGENCIA 10
2.2 MONOCULARES 11
2.2.1 OCLUSIÓN 11
2.2.2 TAMAÑO RELATIVO 11
2.2.3 PERSPECTIVA LINEAL 11
2.2.4 TAMAÑO DE IMAGEN RETINAL 11
2.2.5 PERSPECTIVA ATMOSFÉRICA 11
2.2.6 GRADIENTE DE TEXTURA 12
2.2.7 SOMBREADO 12
2.2.8 PARALAX DE MOVIMIENTO 12
2.3 SEÑALES BINOCULARES 12
2.3.1 PARALAX BINOCULAR 12
2.3.2 ESTEREOPSIS 13
2.4 SEÑALES DE PROFUNDIDAD EN REALIDAD AUMENTADA 13
2.4.1 SEÑALES TRADUCIBLES 13
2.4.2 SEÑALES SIMULABLES 13
2.4.3 SEÑALES NO TRADUCIBLES 14
3 ESTADO DEL ARTE 15
3.1 SIMULACIÓN DE SEÑALES DE PROFUNDIDAD 15
3.2 INFORMACIÓN DERIVADA DEL ENTORNO VIRTUAL 15
3.3 INFORMACIÓN DERIVADA DE ELEMENTOS VIRTUALES 16

4 IMPLEMENTACIÓN Y DISEÑO 17
4.1 EXPLICACIÓN DE LOS MÉTODOS 17
4.1.1 DISTANCE LINE (DL) 17
4.1.2 EARTH LINE (EL) 18
4.1.3 BEEP 18
4.2 TECNOLOGÍAS USADAS 19
4.2.1 UNITY 19
4.2.2 ARCore 19
4.2.3 DISPOSITIVO MÓVIL 20
4.2.4 AUDACITY 20
4.2.5 JUPYTER NOTEBOOK 20
4.2.6 BLENDER 20
4.3 HelloARCore 21
4.3.1 FLUJO BÁSICO 21
4.3.2 CREAR OBJETOS 21
4.4 DISTANCE LINE 22
4.4.1 DISTANCE LINE COMO CLASE 22
4.4.2 CONTROLADOR DE DISTANCE LINE 23
4.4.3 COORDINACIÓN CON HelloARCore 23
4.5 EARTH LINE 24
4.5.1 IMPLEMENTACIÓN 24
4.5.2 COORDINACIÓN CON HelloARCore 24
4.6 BEEP 25
4.6.1 IMPLEMENTACIÓN 25
4.6.2 ACTIVAR Y DESACTIVAR EL SONIDO 25
4.6.3 COORDINACIÓN CON ARCore 26
4.6.4 DIAGRAMAS DE CLASE 26
4.7 PRUEBAS 33
4.7.1 PROCESOS PREVIOS 33
4.7.2 TRANSCURSO DE LA PRUEBA 33
4.7.3 FINALIZAR 33
5. PRUEBAS DE USUARIO 34
5.1 MOTIVACIÓN 34
5.2 MÉTODOS 34
5.2.1 DISTANCE LINE (DL) 34

5.2.2 EARTH LINE (EL) 35
5.2.3 BEEP 35
5.3 PRUEBAS 35
5.3.1 PRUEBA SINGULAR 36
5.3.2 PRUEBA COMPARATIVA 37
5.3.3 PRUEBA RELATIVA 38
5.4 DESARROLLO DE LAS PRUEBAS DE USUARIO 39
5.5 RESULTADOS 40
5.5.1 RESULTADO PRUEBA COMPARATIVA 40
5.5.2 RESULTADO PRUEBA SINGULAR 41
5.5.3 RESULTADO PRUEBA RELATIVA 43
5.6 CONCLUSIONES 44
6.ANEXO 45
6.1 FORMULARIOS 45
6.1.1 FORMULARIO INCIAL 45
6.1.2 FORMULARIO DE LAS PRUEBAS 46
6.2 CÓDIGO DE JUPYTER NOTEBOOK 52
6.2.1 LIBRERÍAS 52
6.2.2 PREPROCESAMIENTO DE LOS DATOS 52
6.2.3 CÁLCULO DE ERRORES 54
6.2.4 CÁLCULO DE LA MEDIA DEL TIEMPO 55
6.3 DATOS 56
6.3.2 PRUEBA COMPARATIVA 56
6.3.3 PRUEBA SINGULAR 57
6.3.4 PRUEBA RELATIVA 58

AGRADECIMIENTOS

Gracias a Iñigo Ezcurdia y a Sebastián Marichal por ayudarme hasta el último momento.

A todas las personas que se presentaron voluntarias para las pruebas de usuario.

También a mis compañeros de laboratorio y a Josune Sorbet Molina.

1. INTRODUCCIÓN

1.1 RESUMEN

El objetivo de este trabajo es desarrollar y estudiar distintas formas de representar la profundidad en la realidad aumentada.

Para ello se van a implementar tres métodos que, presumiblemente, ayudarán a la persona usuaria a percibir mejor la distancia que la separa de uno o varios objetos virtuales. Adicionalmente se van a diseñar e implementar tests mediante los que poner a prueba los distintos métodos desarrollados y evaluar su eficiencia, usabilidad y facilidad de uso.

Con los datos recogidos, y tras analizarlos, podremos sacar conclusiones acerca del rendimiento de los distintos métodos.

1.2 PALABRAS CLAVE

- Realidad aumentada.
- Pruebas de usuario.
- Percepción de la profundidad.

1.3 OBJETIVO

En este trabajo hemos diseñado e implementado tres métodos distintos de ayuda al usuario para la interacción y percepción de las profundidades y distancias de objetos virtuales ubicados mediante realidad artificial.

Así mismo hemos diseñado distintos entornos que simulan funcionalidades reales de aplicaciones AR. La intención es observar, medir y evaluar el desempeño de las distintas tecnologías en tareas que abarcan diferentes usos prácticos de la realidad aumentada.

El objetivo final es extraer conclusiones acerca de los diferentes métodos, sus usos más apropiados, sus límites y posibles aplicaciones.

1.4 HISTORIA DE LA REALIDAD AUMENTADA

1.4.1 DEFINICIÓN

Realidad aumentada, o AR, es un término acuñado en 1990 por Tom Caudell **¡Error! No se encuentra el origen de la referencia.** que hace referencia al conjunto de tecnologías que permiten a un usuario visualizar el mundo real a través de un dispositivo que añade elementos virtuales.

1.4.2 ORÍGENES

La primera constancia que se tiene sobre el concepto de AR fue en '*La llave maestra: un cuento de hadas eléctrico*', una novela escrita en 1901 por L. Frank Braum, autor de '*El mago de Oz'.* En él, el protagonista adquiere unas gafas llamadas '*Marcador de carácter*' que permiten al portador ver la personalidad de las personas flotando sobre sus cabezas en forma de letras.

En 1957 Morton Heilig inventó el *Sensorama*, una máquina capaz de emitir estímulos visuales, vibraciones, sonidos y olores a un usuario. Se trataba de un dispositivo puramente mecánico, por lo que no se le puede considerar el primer dispositivo de AR.

La primera instancia reconocida de realidad aumentada fue creada en 1968 por Ivan Sutherland. El sistema '*head-mounted three dimensional display*' **¡Error! No se encuentra el origen de la referencia.**también conocido como '*Espada de Damocles'* era capaz de recrear imágenes suministradas por un programa informático y visualizarlas en un estetoscopio. La perspectiva mostrada por el software dependía de la posición de la mirada del usuario, por lo que era necesario un sistema de tracking apuntando a los ojos de la persona. El tamaño del dispositivo lo hacía inmanejable, por lo que era necesario un brazo robótico para suspenderlo del techo.

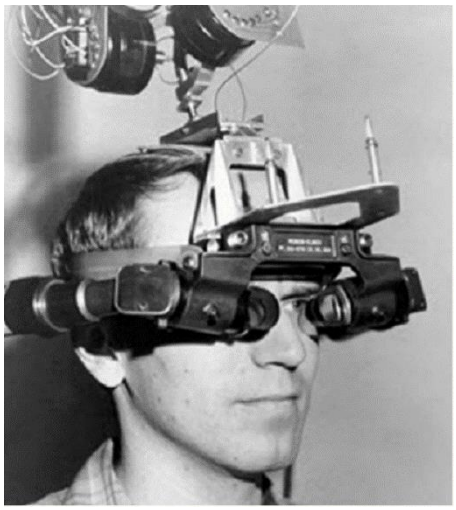

Ilustración 1 Experimento de la espada de Damocles (1)

El siguiente avance vino de parte de Myron Krueger en 1975 a través de una exposición conocida como '*Videoplace'* [3] donde los usuarios podían interactuar y manipular directamente el medio virtual en tiempo real.

No será hasta 1992 cuando veremos el primer sistema de realidad aumentada plenamente funcional de la mano de Louis Rosenberg. '*Virtual Fixture'* **¡Error! No se encuentra el origen de la referencia.**, que así se llamaba el proyecto, estaba formado por dos robots controlados por un exoesqueleto puesto en el tren superior de un usuario. El sujeto debía llevar a su vez un par de lupas a modo de gafas que reducían su campo de visión de modo que pareciese que los dos brazos robóticos ocupaban el lugar de los suyos propios. El sistema utilizaba una serie de imágenes generadas por ordenador diseñadas para ayudar al usuario a llevar a cabos sus tareas.

A partir de la década de los 90 los avances en la realidad aumentada se harán cada vez más frecuentes. De todos ellos, cabe mencionar que de 1996 a 1998 la NHL (National Hockey League) implementó una tecnología llamada '*FoxTrax' (*más conocida como '*Glow Puck'*) en las retransmisiones de los partidos de hockey que hacía brillar el disco de un color azul, si el disco superaba los 112 km/h el color cambiaba a rojo. Esto marca el primer uso de realidad aumentada dirigido al público en masa, y aunque la tecnología se dejó de usar en 1998 dado su impopularidad

entre los espectadores, la NFL contrató al equipo responsable para desarrollar una tecnología capaz de mostrar la línea de 'down' en un partido, lo cual se sigue usando hasta hoy en día.

Ilustración 2 Glow Puck (2)

Ilustración 3 Linea de down (3)

1.4.3 ESTADO ACTUAL

Hoy en día muchas barreras que frenaban el desarrollo de la realidad aumentada ya no existen gracias al desarrollo de la tecnología. La renderización de gráficos 3D en tiempo real es posible en cualquier dispositivo y la disponibilidad de las cámaras de los móviles hace que el requerimiento de hardware se pueda cumplir fácilmente.

Algunos usos mainstream del AR son los siguientes:

- Filtros de Instagram y otras aplicaciones como FaceApp.
- Pancartas publicitarias en los estadios deportivos.
- Aplicación de Ikea para visualizar productos.
- Videojuegos como PokemonGo.
- Letreros y maquetas en informativos u otros programas de televisión.
- Ayudas visuales en deportes como la línea de down o la línea de fuera de juego.

No obstante, la realidad aumentada también ha permeado en varios aspectos de la industria tales como:

- Asistencia remota de expertos [4] **¡Error! No se encuentra el origen de la referencia.**
- Asistencia en tareas logísticas **¡Error! No se encuentra el origen de la referencia.**.
- Líneas de ensamblaje [7] .

En todos estos casos el AR se utiliza como herramienta de apoyo, mejorando el rendimiento o facilitando tareas mediante una comunicación entre la interfaz y el mundo físico más orgánica.

1.4.4 DESARROLLO Y USOS FUTUROS

Hoy en día hay proyectos e ideas que prometen avances a corto o medio plazo que de aquí a pocos años veremos hechos realidad.

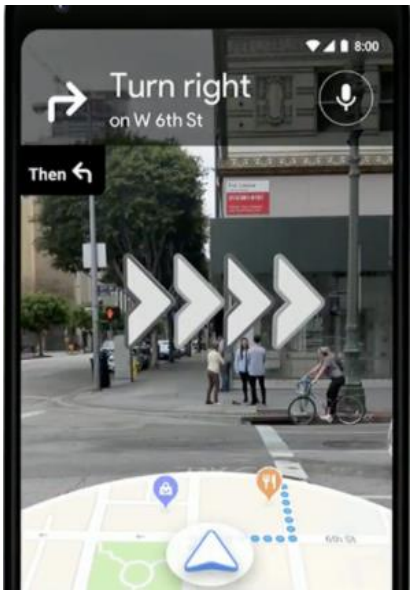

Ilustración 2 Realidad aumentada en GoogleMaps (4)

Una propuesta popular en la industria del retail es la de desarrollar una app que te permita verte con otra ropa [13] [14]. Esta idea ha ido ganando cada vez más interés en los últimos años, principalmente debido a la situación actual con el CoVid y todas las restricciones que ello conlleva.

GoogleMaps permite desde hace un tiempo complementar la información ofrecida por GPS junto con imágenes de los edificios de tu entorno para dar información más precisa de la localización. Además, la compañía está implementando una interfaz de realidad virtual que muestre señales integradas en el entorno. Estas señales podrían aportar información extra como el nombre de las calles o la dirección que hay que seguir para llegar a algún destino.

2. PERCEPCIÓN DE LA PROFUNDIDAD

Para entender los problemas a los que se enfrenta la realidad aumenta en cuanto a percepción de la profundidad, primero tenemos que comprender los procesos físicos y psicológicos por los que somos capaces de distinguir diferentes distancias en la vida real [8]**¡Error! No se encuentra el origen de la referencia.**

Existen tres tipos distintos de señales que indican profundidad:

1.- Oculomotoras: Aquellas basadas en la posición de los ojos y la tensión producida al tener que mirar un objeto.

2.- Monoculares: Que dependen de la visión de un solo ojo.

3.- Binoculares: Que dependen de la visión de ambos ojos.

2.1 SEÑALES OCULOMOTORAS

2.1.1 ACOMODACIÓN

Es la tensión del músculo que cambia la distancia focal de la lente de los ojos. No es una señal muy efectiva, y solo funciona en distancias de menos de 2 metros.

2.1.2 CONVERGENCIA

La diferencia de dirección de los ojos que se produce cuando observamos un objeto cercano. Al igual que la acomodación, solo es fiable en distancias cortas de menos de 10 metros.

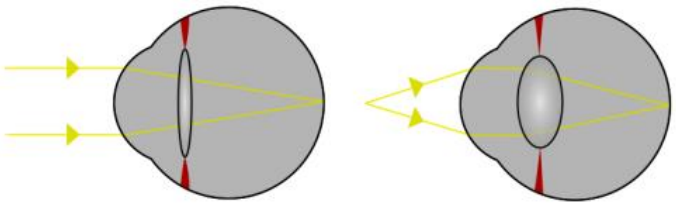

Ilustración 5 Acomodación del ojo (5)

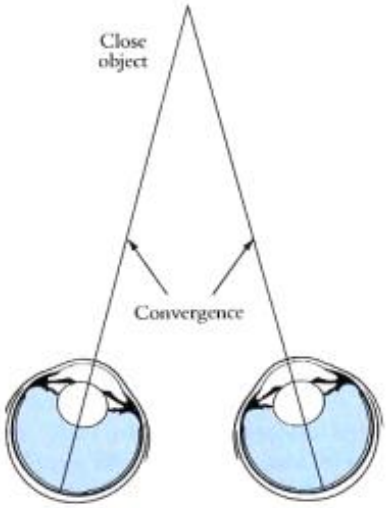

Ilustración 6 Convergencia de los ojos (5)

2.2 MONOCULARES

Existen a su vez dos tipos de señales monoculares: aquellas que se pueden extraer de imágenes fijas (como cuadros o fotografías) o las que dependen del movimiento.

Al primer grupo pertenecen la oclusión, tamaño relativo, perspectiva lineal, tamaño de imagen retinal, perspectiva atmosférica, gradiente de textura y el sombreado.

Al segundo grupo pertenece únicamente el parallax de movimiento.

2.2.1 OCLUSIÓN

Es el fenómeno que se produce cuando un elemento oculta a otro que está más lejos.

En la figura 1 de la imagen 7, el rectángulo A ocluye al B. En la figura 2, el rectángulo B ocluye al A.

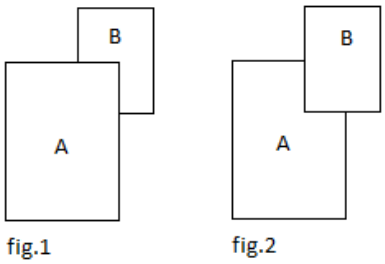

Ilustración 7 Ejemplo de oclusión (6)

2.2.2 TAMAÑO RELATIVO

Cuando conocemos el tamaño de dos objetos, podemos juzgar la distancia a la que están por la diferencia de tamaño que tienen.

2.2.3 PERSPECTIVA LINEAL

Las líneas paralelas parecen juntarse en el horizonte, a este fenómeno se le conoce como perspectiva lineal.

2.2.4 TAMAÑO DE IMAGEN RETINAL

Cuando conocemos el tamaño exacto de un objeto, nuestro cerebro compara el tamaño de la imagen que llega a nuestra retina con el tamaño real. Esto aporta información sobre la distancia a la que está el objeto.

2.2.5 PERSPECTIVA ATMOSFÉRICA

Los objetos muy distantes (como edificios lejanos o montañas) parecen difuminados, borrosos y con un tinte azulado. Este efecto (que se produce por las partículas de polvo y aire entre nuestros ojos y el elemento observado) también nos dice lo lejos que está algo.

2.2.6 GRADIENTE DE TEXTURA

Cuanto más cerca estamos de un objeto, mejor vemos su superficie y comprendemos su textura. Conforme nos alejamos, la textura se va aplanando, por lo que asociamos superficies suaves y homogéneas con elementos lejanos.

Ilustración 8 Ejemplo de Perspectiva atmosférica (7)

Ilustración 94 Ejemplo de gradiente de textura (8)

2.2.7 SOMBREADO

Conociendo la localización de la fuente de luz que ilumina una escena, podemos extraer información muy precisa a partir de las sombras que producen los objetos.

2.2.8 PARALAX DE MOVIMIENTO

Cuando nos movemos percibimos que los objetos que están más cerca nuestra se mueven mucho más deprisa que los que están alejados.

Un ejemplo de esto es en un trayecto en coche, cómo percibimos lo rápido que se mueven las señales de tráfico cuando las cruzamos respecto al sol, que parece inmóvil.

2.3 SEÑALES BINOCULARES

2.3.1 PARALAX BINOCULAR

Es la diferencia en la forma que tenemos de percibir los objetos desde distintas posiciones. Es la señal más importante y que más información aporta.

2.3.2 ESTEREOPSIS

No es tanto una señal como un fenómeno que sucede en nuestro cerebro.

Cada uno de nuestros ojos aporta una imagen bidimensional ligeramente distinta de lo que estemos observando. Nuestro cerebro es capaz de recomponer estas dos imágenes en una sola perspectiva tridimensional. A este proceso se le conoce como estereopsis.

2.4 SEÑALES DE PROFUNDIDAD EN REALIDAD AUMENTADA.

Uno de los objetivos de la realidad aumentada es poder llegar a imitar estas señales, ya que son la forma más efectiva e intuitiva de transmitir la distancia que conocemos. Esto supone un problema, ya que, aunque algunas se traducen perfectamente a un ambiente virtual, otras son muy costosas o incluso imposibles de implementar.

Vamos a explicar poco a poco los retos que supone traducir los métodos físicos por los que recibimos la información visual a un ambiente simulado.

2.4.1 SEÑALES TRADUCIBLES

Algunas de las señales pueden ser simuladas en un entorno virtual que respete las leyes de la física.

El tamaño relativo, la perspectiva lineal, el paralax de movimiento y el tamaño de imagen retinal responden a la propiedad de los objetos de volverse más pequeño conforme se alejan. Los motores 3D ya representan de forma correcta esta característica, por lo que cualquier aplicación desarrollada en un motor actual será capaz de transmitir estas señales de forma automática.

La oclusión entre elementos virtuales también es una funcionalidad básica de los motores 3D.

Muchos motores actuales disponen también de la capacidad de simular luces y sombras, aunque se trata de una tecnología computacionalmente costosa. No obstante, para poder representar fielmente la sombra que producen los objetos virtuales, la tecnología utilizada debe ser capaz de reconocer las fuentes de luz del ambiente en el que está simulando los objetos. Esto supone aún más coste computacional, pero como hemos visto, el sombreado es una señal ambiental muy potente a la hora de traducir información espacial.

El gradiente de textura es un caso particular, ya que el motor 3D es capaz de renderizar texturas de forma preciso incluso de objetos lejanos. No obstante, cuanto más pequeño sea el objeto en pantalla, menos píxeles ocupará la imagen, y menos definida quedará la superficie del objeto.

2.4.2 SEÑALES SIMULABLES

Acabamos de ver que la oclusión entre objetos virtuales sucede de forma automática en los motores 3D, el verdadero reto es conseguir que respondan de forma realista a los elementos reales.

Hay dos formas de conseguir esto, la primera es realizar un modelado completo del ambiente físico donde se vaya a realizar la prueba de realidad aumentada, de esta forma, el sistema tendrá equivalentes virtuales de todos los objetos reales y respetará las físicas; la segunda opción es realizar un mapa de profundidad del ambiente a tiempo de ejecución. Es una opción más complicada, pero desarrolladores como GoolgeDevelopers ya lo están realizando **¡Error! No se encuentra el origen de la referencia.**.

La perspectiva atmosférica es un evento que se produce por el aire y el polvo, por lo que sería imposible simularlo (o el coste computacional de simular partículas de aire sería tan alto que lo volvería inviable). No obstante, hay soluciones que podrían producir la misma sensación, como aplicar filtros en objetos muy lejanos.

2.4.3 SEÑALES NO TRADUCIBLES

La acomodación y la convergencia son dos fenómenos imposibles de recrear. Ambos procesos suceden cuando los ojos enfocan un objeto, y en el caso de la realidad aumentada, siempre estamos enfocando una pantalla. Como los objetos simulados siempre están más allá de la pantalla para ser visibles, no es posible recrear la información.

Dicho esto, existen diferencias que dependen del dispositivo de realidad aumentada que se está usando. Los dispositivos móviles presentan toda su información en una única pantalla, independientemente de lo lejos que estén en la simulación los objetos mostrados. Esto quiere decir que, a la hora de observarlos, el punto de convergencia de los ojos está a la misma distancia en toda la imagen.

Las gafas de realidad aumentada, por otro lado, proyectan una imagen en cada ojo y nuestro cerebro se encarga de juntarlo después, por lo que el punto de convergencia de la mirada sí que depende de la distancia del objeto virtual.

En nuestro caso, como vamos a realizar las pruebas con un dispositivo móvil, concluiremos que no es posible traducir esta información.

3 ESTADO DEL ARTE

3.1 SIMULACIÓN DE SEÑALES DE PROFUNDIDAD

Hemos visto los mecanismos mediante los cuales somos capaces de percibir la profundidad de los objetos en el mundo real. También hemos visto que existe información que no puede ser traducida a un entorno virtual.

También existen limitaciones que hacen imposible aplicar todas las tecnologías disponibles en un único entorno de desarrollo, y en caso de que se pudiera, el coste computacional sería enorme.

Por este motivo, los trabajos más recientes sobre la percepción de la profundidad se centran más en el uso de señales selectivas, con la idea de evaluar los mejores usos para cada una y el desempeño en distintos ambientes.

Por lo general, existen señales que aportan información más precisa que otras. En evaluaciones sobre una misma prueba de percepción de distancia **¡Error! No se encuentra el origen de la referencia.**los mejores resultados se daban cuando los objetos virtuales emitían sombra, mientras que otras señales como el gradiente de textura y la perspectiva atmosférica no eran capaces de expresar información precisa.

Cipiloglu et al. [11] exploran la posibilidad de elegir dinámicamente las señales dependiendo de las características de cada prueba, de forma que solo se apliquen las que aportan mejor información. De esta forma evitamos un uso excesivo de la capacidad de computación del sistema. Este acercamiento corre el riesgo de que los distintos componentes aporten información contradictoria si no están bien integrados entre sí.

En conclusión, aunque las tecnologías que tratan de simular las señales de profundidad son cada vez más fiables, no siempre es necesario o justificable su uso, por lo que elegir las más eficaces parece ser la mejor respuesta por el momento.

3.2 INFORMACIÓN DERIVADA DEL ENTORNO VIRTUAL

Existe un acercamiento totalmente opuesto al anterior: aprovechar las características del entorno virtual para crear nuevas referencias que aporten información al usuario. De esta forma podemos crear nueva información a partir de elementos distintos a las señales que hemos estudiado.

En "Depth Perception in Projective Augmented Reality: An Evaluation of Advanced Visualization Techniques" **¡Error! No se encuentra el origen de la referencia.**realizan pruebas en las cuales la información se transmite en los propios objetos, de forma que su representación es distinta dependiendo de la altura a la que están. También cambian el propio entorno en el que tiene lugar

la prueba e incluso añaden nuevos elementos virtuales que aportan más información sobre los objetos.

Concluyen el estudio con los dos métodos con mejores resultados. Uno modifica el color de los objetos según su altura (fig.1). Otro añade dos elementos virtuales más a la escena, una línea desde la base del objeto al suelo y las sombras que proyectarían (fig.2).

Aunque es cierto que las sombras proyectadas son una de las señales físicas que hemos visto, el resto de los componentes de las pruebas son puramente virtuales.

No está claro si la información añadida virtualmente es más eficiente o eficaz que las señales físicas reales, pero todo parece indicar que son igual de válidas a la hora de transmitir la profundidad de los objetos.

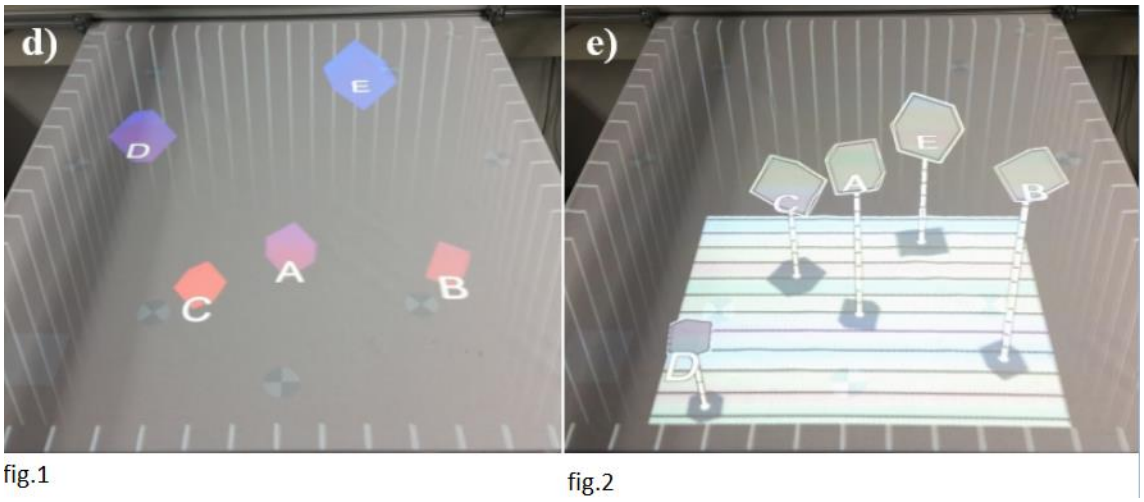

fig.1 fig.2

Ilustración 10 Pruebas realizadas en el estudio (9)

3.3 INFORMACIÓN DERIVADA DE ELEMENTOS VIRTUALES

Como acabamos de ver, es posible utilizar elementos virtuales para transmitir información de otros elementos virtuales. Nosotros nos centraremos en este acercamiento.

No utilizaremos ninguna señal de profundidad física (a excepción de las que ya implementa el propio motor 3D) ni modificaremos el entorno en el que se realizan las pruebas. Las únicas herramientas que utilizaremos para transmitir la información serán otros elementos virtuales creados y diseñados para transmitir la sensación de profundidad de maneras distintas.

4 IMPLEMENTACIÓN Y DISEÑO

4.1 EXPLICACIÓN DE LOS MÉTODOS

Hemos diseñado tres métodos distintos, cada uno interactúa con el objeto, el entorno y el usuario de forma distinta. Vamos a dar una breve descripción de cada uno, aunque explicaremos en profundidad los detalles de sus diseños más adelante.

4.1.1 DISTANCE LINE (DL)

Una línea con un extremo pegado al objeto virtual y otro pegado a la cámara. La anchura de la línea varía con la distancia: cuanto más cerca estén los extremos, más gruesa será.

Este método no depende del entorno para comunicar la sensación de profundidad, pero requiere conocer su funcionamiento de antemano.

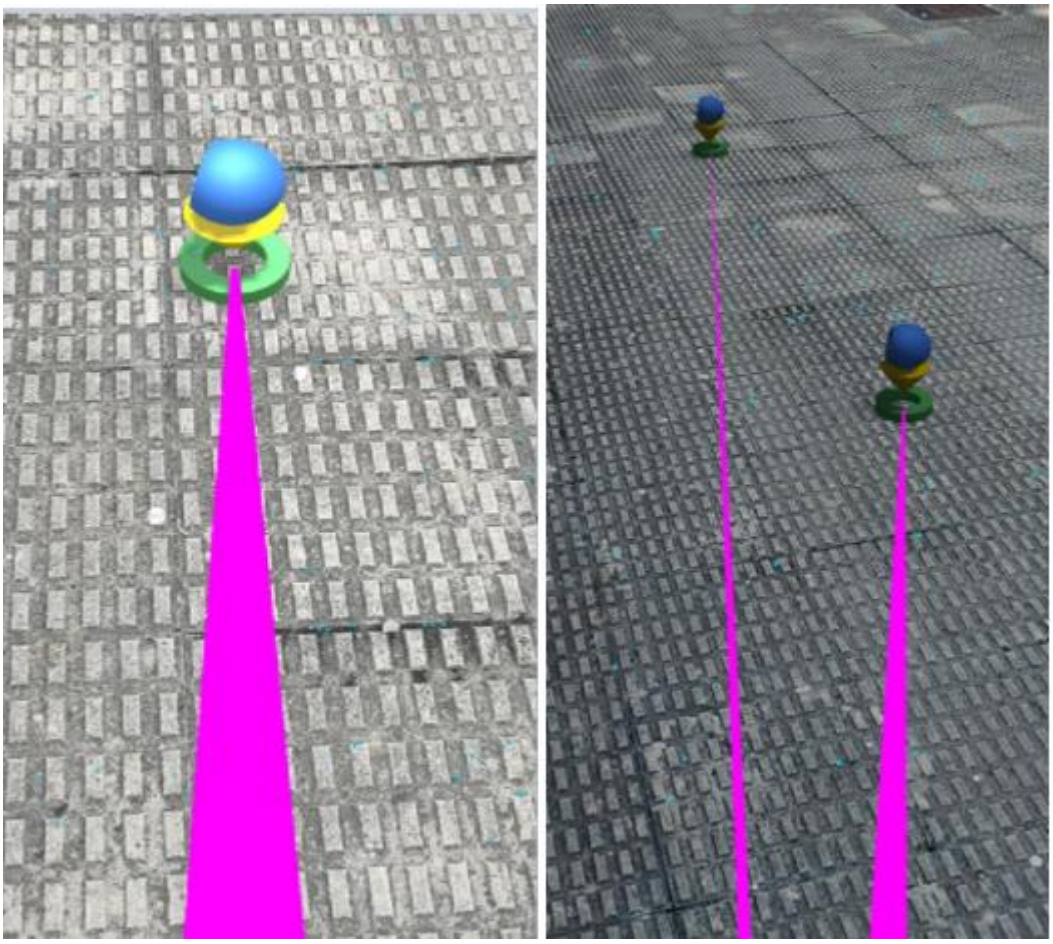

Ilustración 11 Método DL con un y dos objetos

4.1.2 EARTH LINE (EL)

Una línea con un extremo pegado a la base del objeto y otro pegado en el suelo, justo debajo suyo. El grosor no es variable.

Este método se relaciona directamente con el entorno.

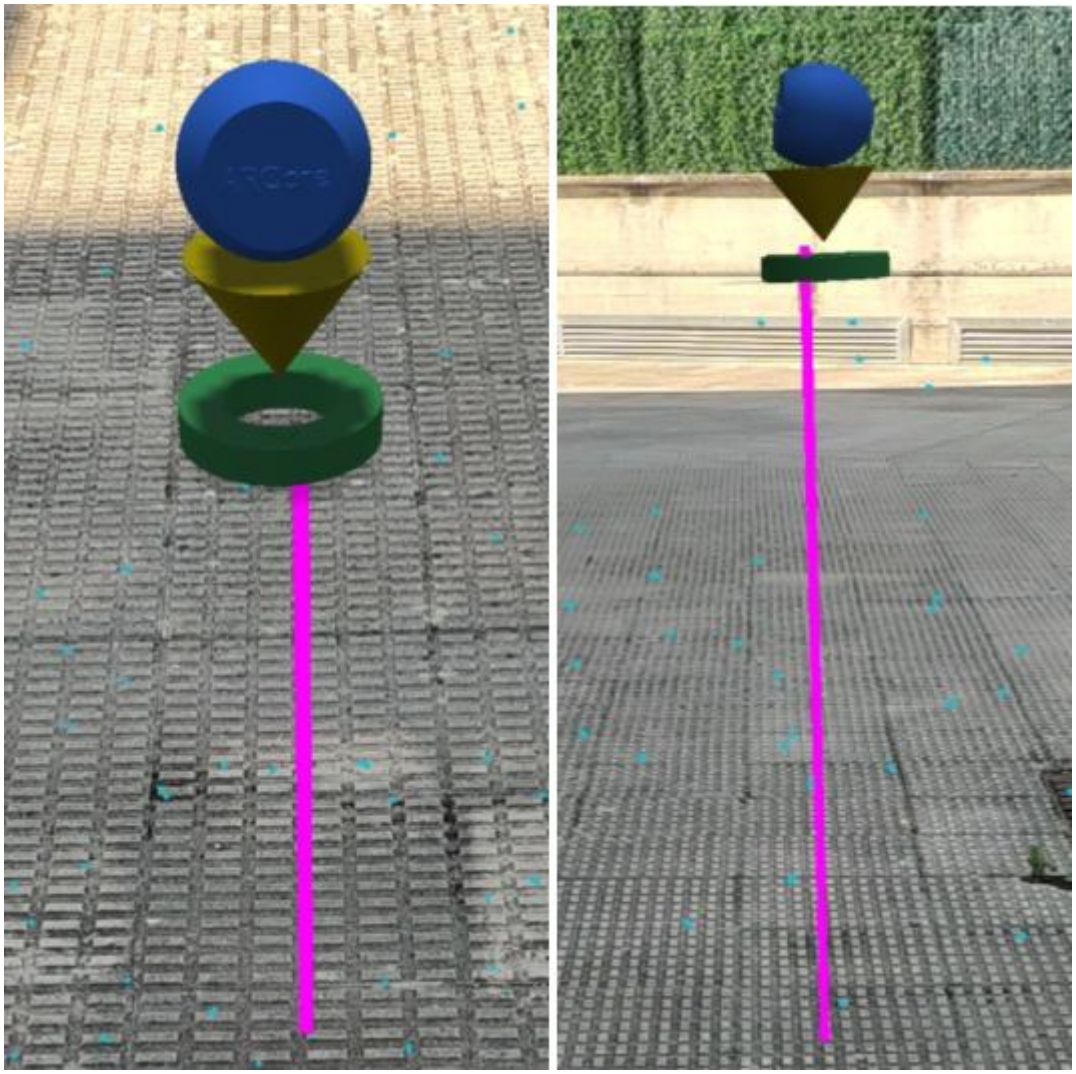

Ilustración 12 Método EL

4.1.3 BEEP

Otorgamos a los objetos la capacidad de emitir sonido de forma realista, es decir, se atenúa con la distancia.

Al contrario que los otros dos, el sonido sí es una señal real que usamos en el mundo físico para relacionarnos con los objetos. Nosotros aprovechamos el entorno virtual para otorgarle la capacidad de emitir sonido a un elemento que a priori no debería poder.

4.2 TECNOLOGÍAS USADAS

4.2.1 UNITY

Unity es un motor de desarrollo 3D utilizado principalmente a la hora de desarrollar videojuegos. Entre otras plataformas, permite desarrollar para Android.

Unity es un software increíblemente potente con una gran cantidad de herramientas a su disposición, nosotros vamos a hacer uso de 3:

- ***Line Renderer:*** Coge como parámetro de entrada un array de 2 o más puntos y dibuja una línea recta entre ellos. Lo usamos en 2 de los 3 métodos y en ambos casos solo necesitamos 2 puntos (la posición del objeto y la posición de la cámara).

  Cabe mencionar que se puede modificar (entre muchos otros parámetros) la anchura de la línea que une los dos puntos.

- ***Audio Source:*** Se puede añadir como componente a un *GameObject* para permitir que reproduzca una (o varias) pistas de sonido. Es posible hacer que la emisión del sonido sea bidimensional o tridimensional, así como modificar la función de emisión del sonido.
- ***Audio Listener:*** Se puede añadir como componente a un *GameObject* para hacer que sea capaz de percibir los sonidos emitidos por otro objeto con un *Audio Source.*

Para evitar problemas de compatibilidad entre diferentes dispositivos y herramientas, hemos utilizado la versión 2018 de Unity.

4.2.2 ARCore

ARCore es una tecnología de desarrollo de realidad aumentada creada por Google. Es accesible desde plataformas como Android Studio o Unity.

El funcionamiento de ARCore se podría dividir en dos partes. Por un lado, es capaz de distinguir puntos de interés en el entorno y monitorizar su movimiento mediante la cámara y los sistemas de movimiento del móvil (acelerómetro, brújula, etc.). Gracias a estos puntos de interés, es capaz de detectar superficies, tanto horizontales como verticales.

Por otro lado, es capaz de crear planos virtuales a partir de las superficies y almacenarlos en memoria. Esto permite al dispositivo “recordar” dónde estaban las superficies, aunque no las estemos mirando en ese momento.

Además de esto, ARCore también proporciona una serie de programas de prueba que sirven como demostración de las utilidades del sistema. Uno en concreto, llamado HelloARCore, permite instanciar pequeños punteros virtuales en las superficies con solo tocar la pantalla.

Hemos utilizado la versión 1.12.2 del sdk para Unity.

4.2.3 DISPOSITIVO MÓVIL

Hemos decidido utilizar un móvil como dispositivo de realidad aumentada por varias razones. Por una parte, es un dispositivo que posee toda la tecnología necesaria para llevar a cabo tanto las funciones básicas (mostrar elementos virtuales en un entorno real) como las algunas más avanzadas (interactividad).

También es un aparato que, por su fácil accesibilidad, es la plataforma más frecuente de realidad aumentada, por lo que los resultados y conclusiones obtenidas serán aplicables a una mayor cantidad de aplicaciones.

Finalmente, su tamaño manejable lo hace perfecto para las pruebas desarrolladas, algunas de las cuales, como veremos más adelante, requieren un grado de maniobrabilidad elevado.

El móvil utilizado para todas las pruebas es un Xiaomi Mi 8.

4.2.4 AUDACITY

Audacity es un programa gratuito de edición de audio.

Hemos usado este programa para crear una pista de audio que utilizaremos en una de las pruebas.

4.2.5 JUPYTER NOTEBOOK

Es un entorno informático interactivo basado en la web que permite crear y ejecutar código cómodamente.

Hemos utilizado este editor para crear programas en Python que procesen los datos recogidos durante las pruebas.

Esta es una lista de las librerías de Python que hemo usado:

- ***Numpy:*** Una librería matemática que permite un uso avanzado y eficiente de vectores y matrices.
- ***Matplotlib:*** Biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays.

4.2.6 BLENDER

Es un programa informático multiplataforma especializado en la creación, iluminación, renderizado, modelado y animación de gráficos 3D.

Aunque el trabajo realizado en Blender fue descartado para las versiones finales del proyecto, en un punto del desarrollo se utilizó para crear nuevos modelos para los objetos virtuales de ARCore.

Finalmente decidimos utilizar los modelos por defecto.

4.3 HelloARCore

Como ya hemos mencionado antes, todas las pruebas están desarrolladas a partir del programa de ejemplo que tiene ARCore. Vamos a explicar cómo funciona, ya que es importante conocer el flujo básico del sistema para entender cómo hemos programado las pruebas.

4.3.1 FLUJO BÁSICO

No vamos a describir el código completo, tan solo vamos a comentar la función Update(), que es la que se llama cada vez que se actualiza la aplicación, y la que lleva a cabo las principales funciones del programa. En orden de ejecución, la función realiza lo siguiente:

1. Si el sistema no ha detectado ningún toque en la pantalla, el programa termina.
2. Si el sistema detecta un toque en la pantalla, pero coincide con la posición de un elemento del interfaz del sistema (botones, sliders, texto…) el programa termina.
3. Proyecta el punto de la pantalla que se ha tocado sobre los planos detectados por el sistema, y guarda ese punto.
    a. Si ese punto está en la parte de detrás de un plano detectado, el programa termina
    b. Si el punto coincide con la superficie de un plano detectado, crea un objeto en ese punto

4.3.2 CREAR OBJETOS

Es importante explicar cómo se crean los objetos en ARCore.

Cuando se toca la pantalla y aparece un objeto, en realidad se están creando dos objetos a la vez. El primero, es un *GameObject*, un tipo de clase instanciable más básico con el que trabaja Unity. El segundo es un *Anchor* (ancla), una clase específica de ARCore.

El *GameObject* es la clase por defecto de Unity. Para crear uno hay que pasar como parámetros de entrada el modelo del objeto que queremos crear, la posición donde lo queremos crear y la rotación en la que queremos que se cree. Unity se encarga de virtualizarlo.

Un *Anchor* es una clase de ARCore que se encarga de fijar un punto en el espacio, pero al contrario que Unity, que sólo reconoce coordenadas de un espacio simulado, un *Anchor* es capaz de rastrear un punto del espacio real. Incluso aunque el usuario se mueva o no esté siendo apuntado por la cámara. Es el propio ARCore el que se encarga de traducir el sistema de referencias de posiciones del espacio real en coordenadas virtuales de Unity. De esta forma, si al crear un *GameObject* lo asignamos a la posición de un *Anchor*, dará la sensación de que está en un punto fijo en el espacio.

4.4 DISTANCE LINE

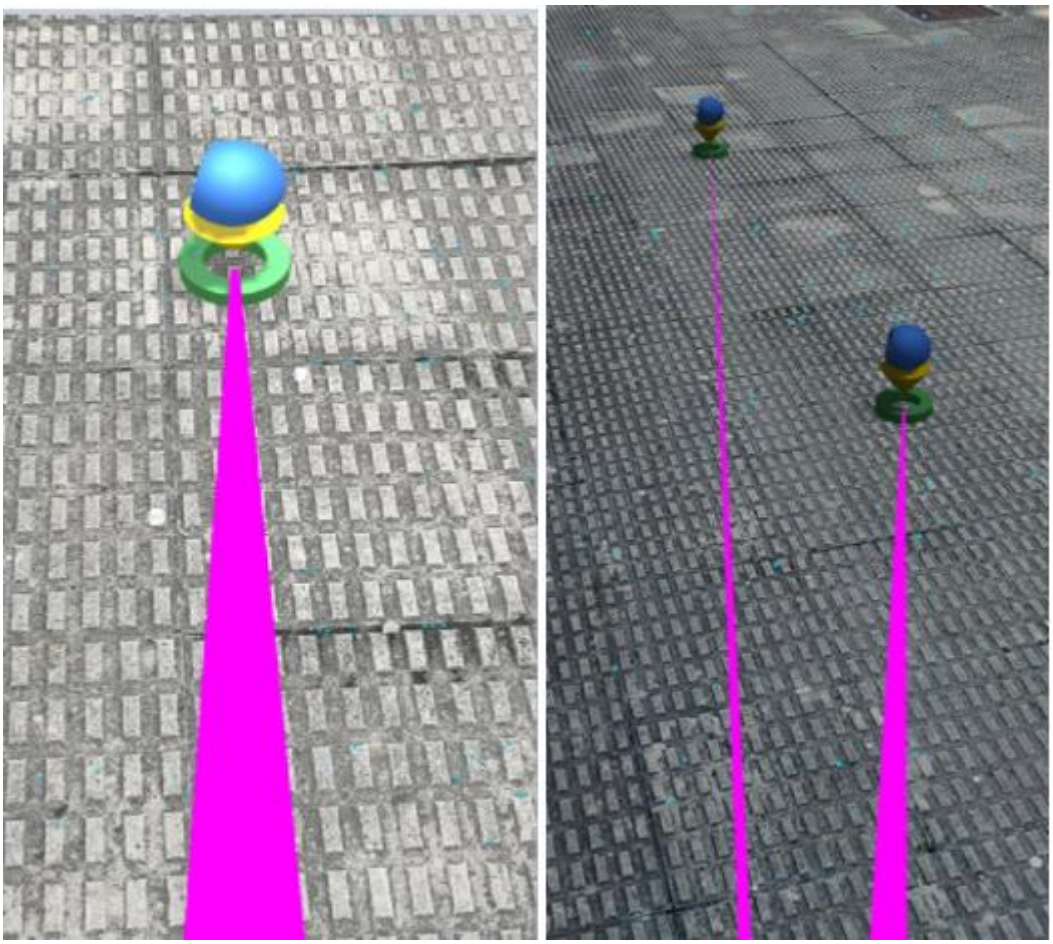

Ilustración 13 Método DL con un y dos objetos

4.4.1 DISTANCE LINE COMO CLASE

El funcionamiento de este método está basado en la herramienta *Line Renderer* de Unity. Esta clase ya nos ofrece las funcionalidades básicas para hacer una línea que se ensanche o encoja según la distancia, no obstante, resulta una función lo suficientemente compleja como para encapsularla en una clase.

La clase consta de los siguientes atributos:

- **Start:** un vector tridimensional que representa la posición en la que se va a crear uno de los extremos del *Line Renderer.*
- **End:** un vector tridimensional que representa la posición en la que se va a crear el otro extremo del *Line Renderer.*
- **Width:** anchura que se le va a asignar al *Line Renderer*, este parámetro se actualizará con cada frame.
- **Anchura_max:** anchura máxima que puede tener.
- **Anchura_min:** anchura mínima que puede tener.
- **D_max:** distancia máxima a partir de la cual la anchura deja de disminuir.

- **D_min:** distancia mínima a partir de la cual la anchura deja de aumentar.

Hay dos funciones que calculan el grosor de la línea, la primera calcula el grosor según una función lineal:

```
((dist - d_min) / (d_max - d_min)) * (anchura_max - anchura_min) + anchura_min
```

La segunda lo calcula según una función sigmoide:

```
anchura = (dist - d_min) / (d_max - d_min)
anchura = (anchura - 0.5f) * (-10f);
anchura = (float)(1/(1 + System.Math.Pow(System.Math.E,anchura*(-1))));
anchura = anchura * (anchura_max - anchura_min) + anchura_min;
```

Nos hemos decantado por la segunda función. La primera aporta información precisa en todas las distancias, pero la función sigmoide transmite información más precisa en distancias medias (entre 1 y 5 metros), que son las que vamos a utilizar en la mayoría de las pruebas.

Esta clase se actualiza siguiendo los siguientes pasos:

1. Actualiza los extremos del *Line Renderer.*
2. Calcula la distancia entre los extremos.
3. Calcula el grosor de la línea en base a esa distancia.
4. Actualiza el grosor del *Line Renderer.*

4.4.2 CONTROLADOR DE DISTANCE LINE

Resulta sencillo instanciar y trabajar con una sola DL, pero para poder controlar de manera más segura y eficiente las distance line en los métodos en los que tenemos que instanciar dos y tres de ellas, hemos creado también una clase llamada '*DL Controller'.*

Es una clase sencilla que almacena DL en una lista y actualiza todas a la vez. También permite actualizar únicamente uno de los extremos de las líneas, lo cual resulta útil para aquellas pruebas en las que uno de los dos extremos está fijo.

4.4.3 COORDINACIÓN CON HelloARCore

Cada vez que instanciemos un objeto, creamos una DL con uno de los extremos en el centro del objeto, y la otra en la cámara (como ya hemos explicado antes, realmente se pone ligeramente por debajo de la cámara).

Para que se actualicen con el movimiento de los objetos y del usuario, hay que llamar a la función Actualizar() al comienzo de la rutina de Update() del programa.

4.5 EARTH LINE

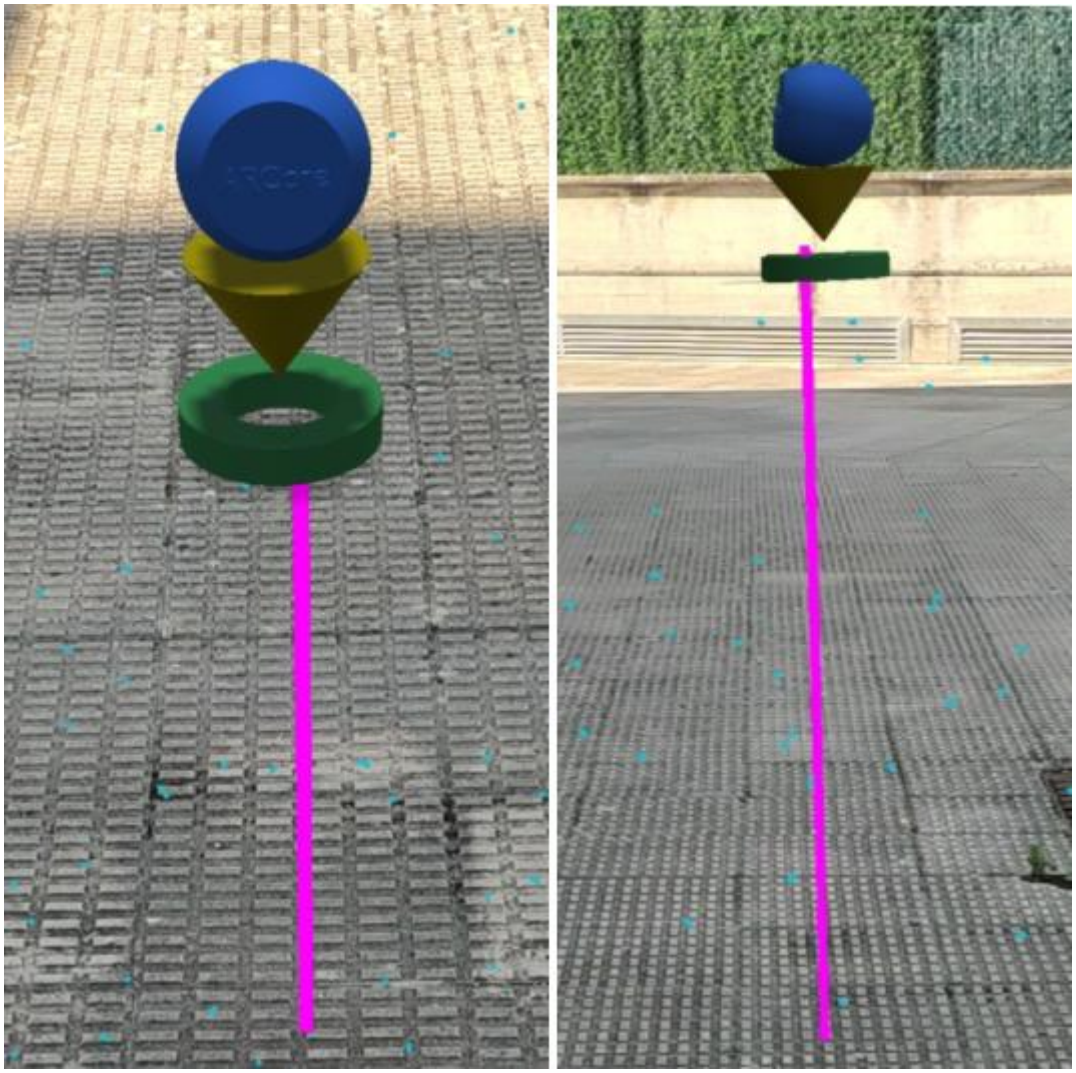

Ilustración 14 Método EL

4.5.1 IMPLEMENTACIÓN

Esta es la clase más sencilla de implementar. Como consta únicamente de una línea con anchura fija, solo hace falta instanciar un *Line Renderer* que una el centro del objeto creado con el punto justo debajo suya.

La posición del objeto podemos conocerla del mismo objeto, ya que es uno de sus atributos (*GameObject.Transform.Position*). Para acceder al punto del suelo solamente hay que cambiar el valor del eje y (la altura) por 0.

4.5.2 COORDINACIÓN CON HelloARCore

Cada vez que instanciemos un objeto, creamos un *Line Renderer* con las coordenadas que acabamos de explicar. En cuanto a la anchura, tiene que ser lo suficientemente gruesa para ser distinguible desde lejos, pero suficientemente fina como para señalar inequívocamente un punto concreto del suelo. Mediante prueba y error nos acabamos decantando por instanciar las líneas con una anchura de 0.01, pero otros valores más grandes también podrían funcionar bien.

4.6 BEEP

4.6.1 IMPLEMENTACIÓN

Para que los objetos puedan emitir sonido tenemos que modificar el modelo por defecto que viene con ARCore. Tenemos que añadir un *Audio Source* como componente, esto permitirá que reproduzcan una pista que le podemos pasar como atributo. Es importante asegurarse de que se está emitiendo sonido en 3D, ya que la intención del método es intentar emular el comportamiento real del sonido.

Hemos creado un archivo de extensión .wav usando Audacity. Se trata de un pitido simple a unos 440 hz. Este será el sonido que emitan los objetos.

Para que el dispositivo pueda captar y retransmitir el sonido que emiten los punteros, tenemos que añadir un *Audio Listener* en la cámara de primera persona de Unity.

4.6.2 ACTIVAR Y DESACTIVAR EL SONIDO

Para completar el método debemos hacer que el Audio Listener de los objetos se enciendan cuando salgan en pantalla y se apaguen cuando no se muestren.

Los GameObject tienen un método llamado IsRenderer() que devuelve True si el objeto está siendo renderizado por la cámara. No obstante, no funciona correctamente en aplicaciones de realidad aumentada, por la forma en la que funciona el sistema, los objetos se mantienen renderizados en todo momento.

La solución que hemos encontrado es proyectar la posición del objeto en el plano de la cámara. Unity ofrece una función llamada WorldToScreenPoint() que realiza esta misma función. Si las coordenadas recibidas son mayores que 0 y menores que el ancho y el alto de la cámara, significa que el objeto está dentro.

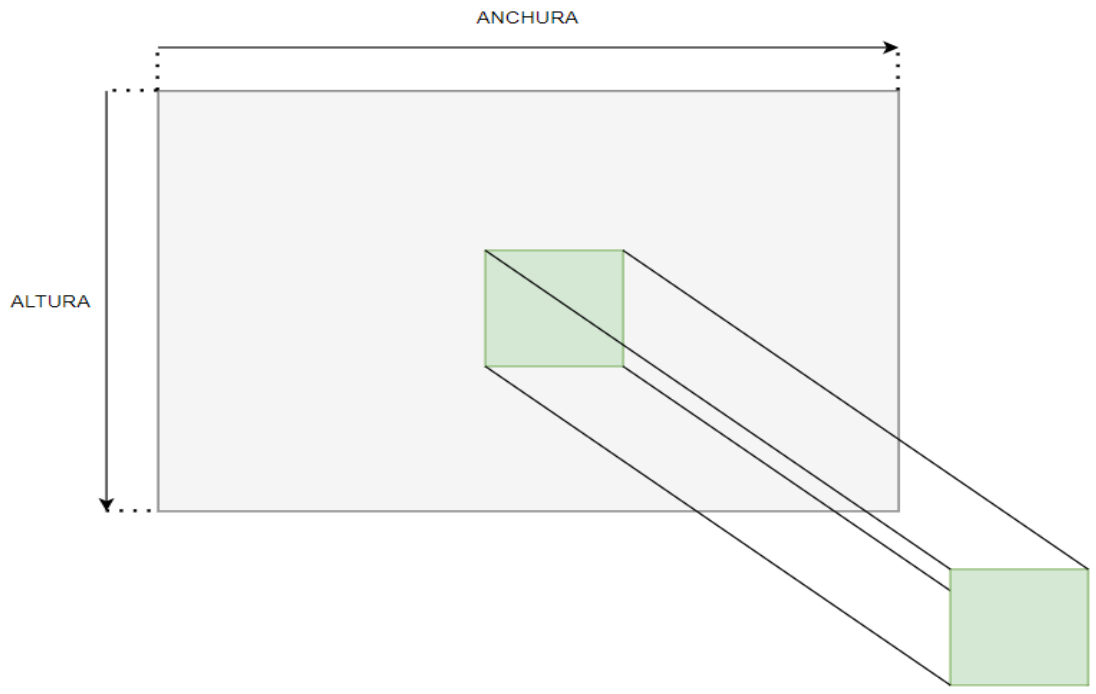

Ilustración 15 Ejemplo de uso de la función WorldToScreenPoint()

4.6.3 COORDINACIÓN CON ARCore

Como hemos modificado el código del modelo de los objetos, no hace falta realizar ningún cambio para que emitan sonido. Tan solo hace falta comprobar en cada frame si el objeto está siendo observado o no, y activar y desactivar el sonido en función de esto.

4.6.4 DIAGRAMAS DE CLASE

Vamos a mostrar los diagramas de clase de cada una de las pruebas.

No vamos a mostrar todos los atributos y los métodos de las clases nativas de Unity ya que hay demasiados componentes mostrarlos empeoraría la comprensión del diagrama. Sí que vamos a añadir aquellos componentes que les añadimos nosotros, es decir, el *LineRenderer* en el caso del EL y el *audioSource* en el caso de Beep.

Tampoco vamos a incluir los elementos que forman parte del HUD (botones, sliders, campos de texto, etc.) ya que, aunque son usados desde la clase HellorARController, no aportan información acerca del verdadero funcionamiento del programa.

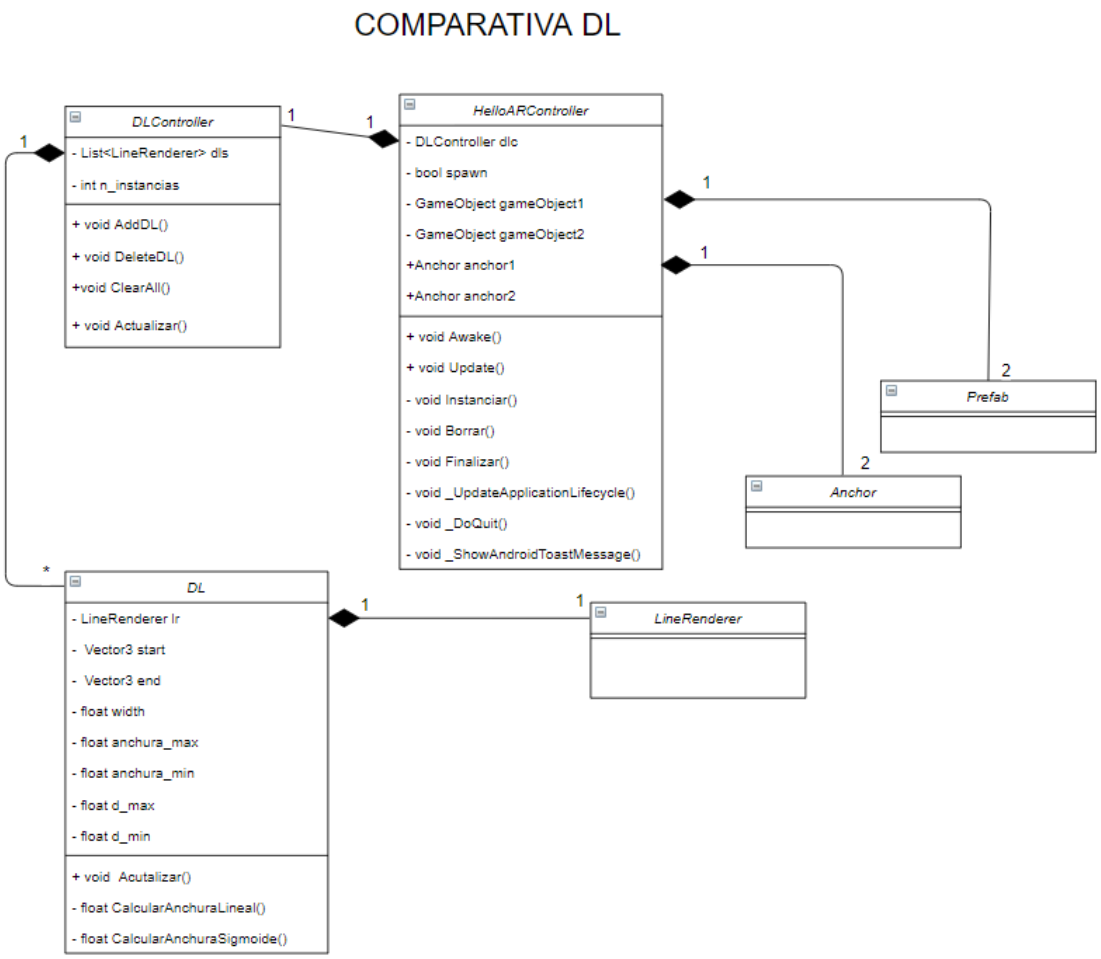

Ilustración 16 Diagrama de clases

COMPARATIVA EL

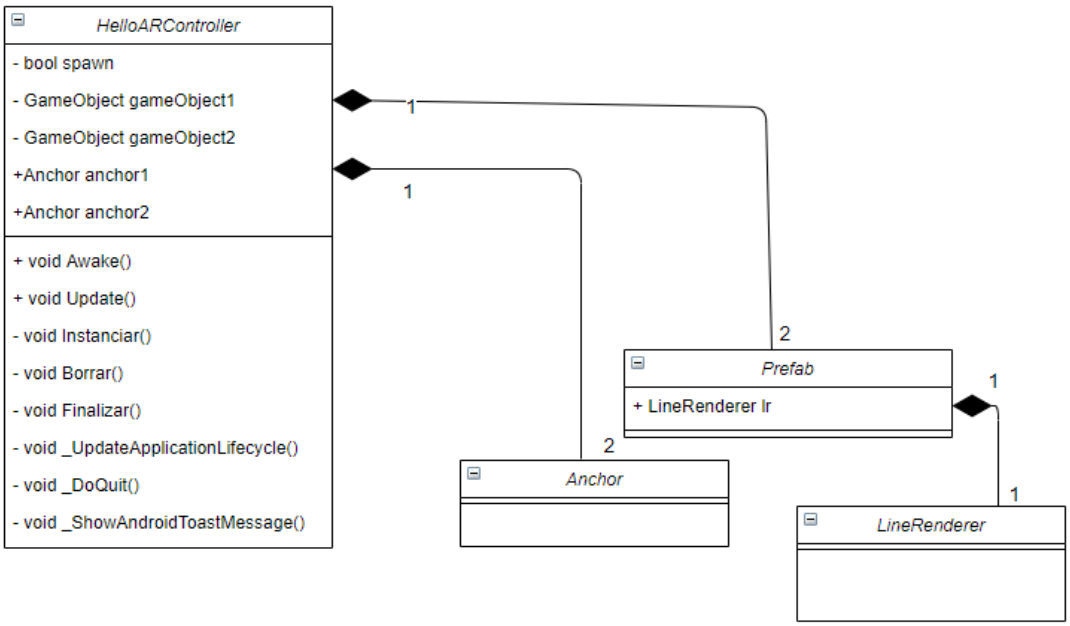

Ilustración 17 Diagrama de clases

COMPARATIVA BEEP

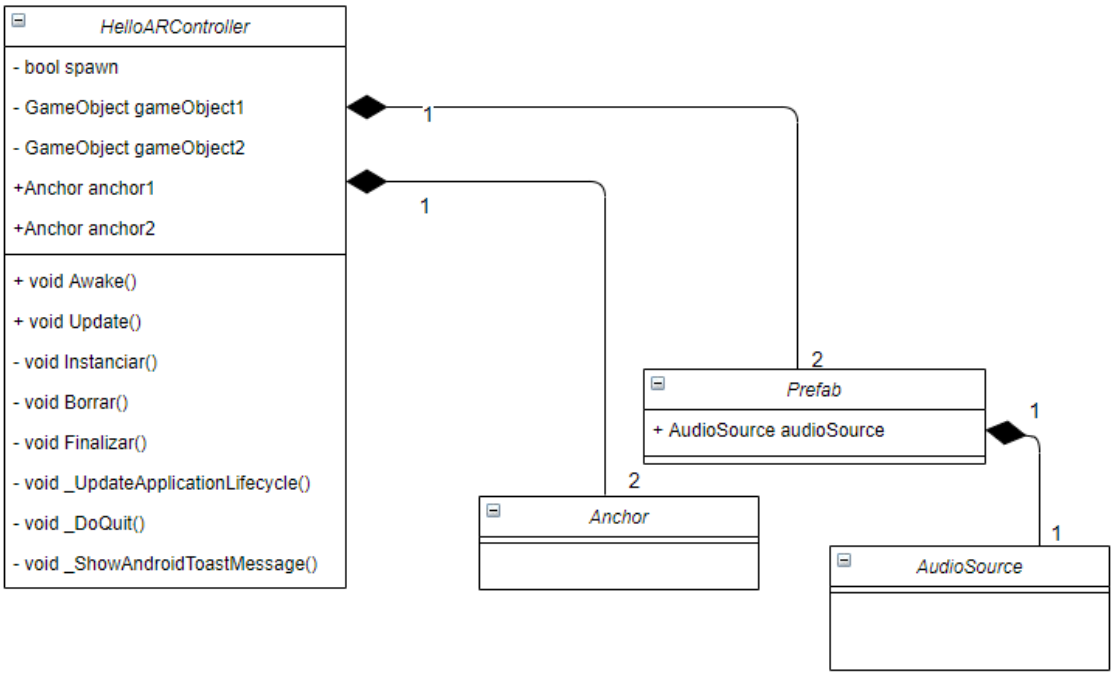

Ilustración 18 Diagrama de clases

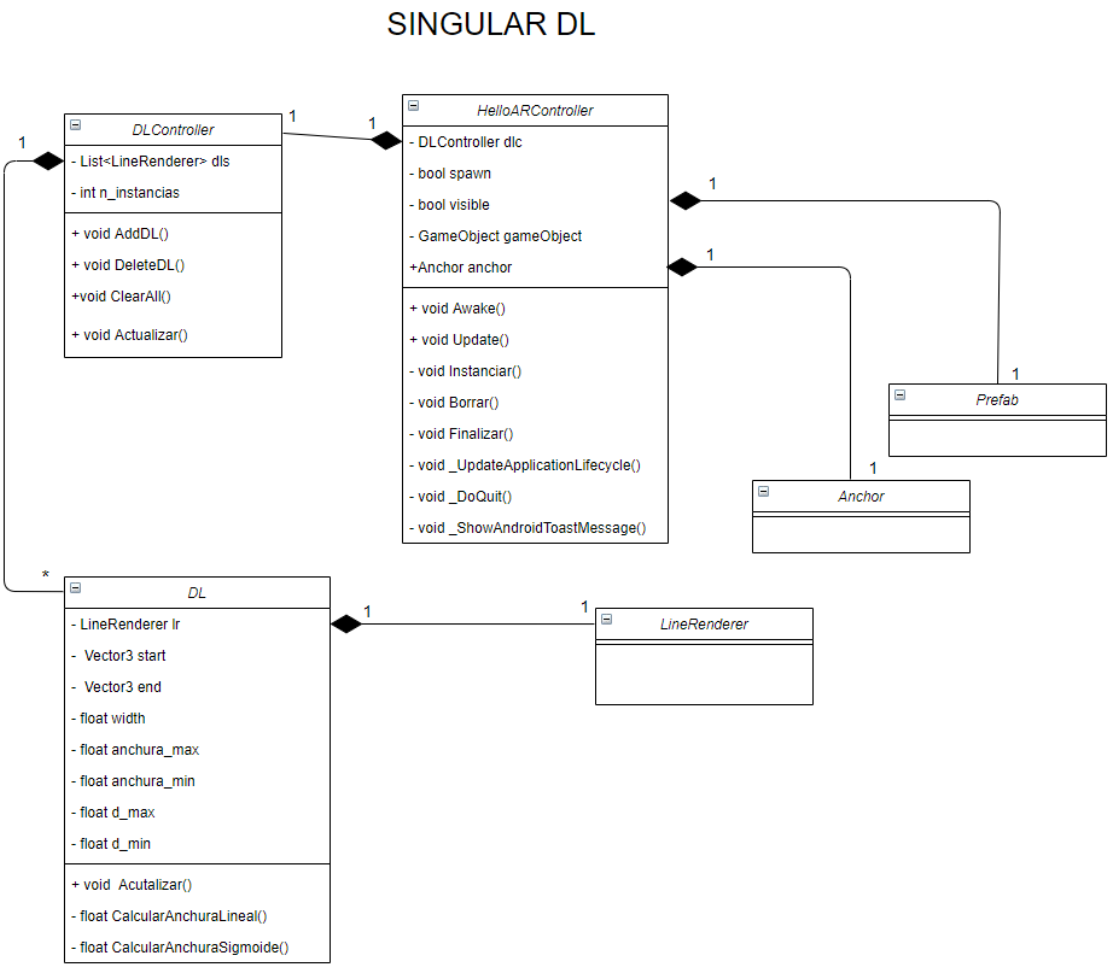

Ilustración 19 Diagrama de clases

SINGULAR EL

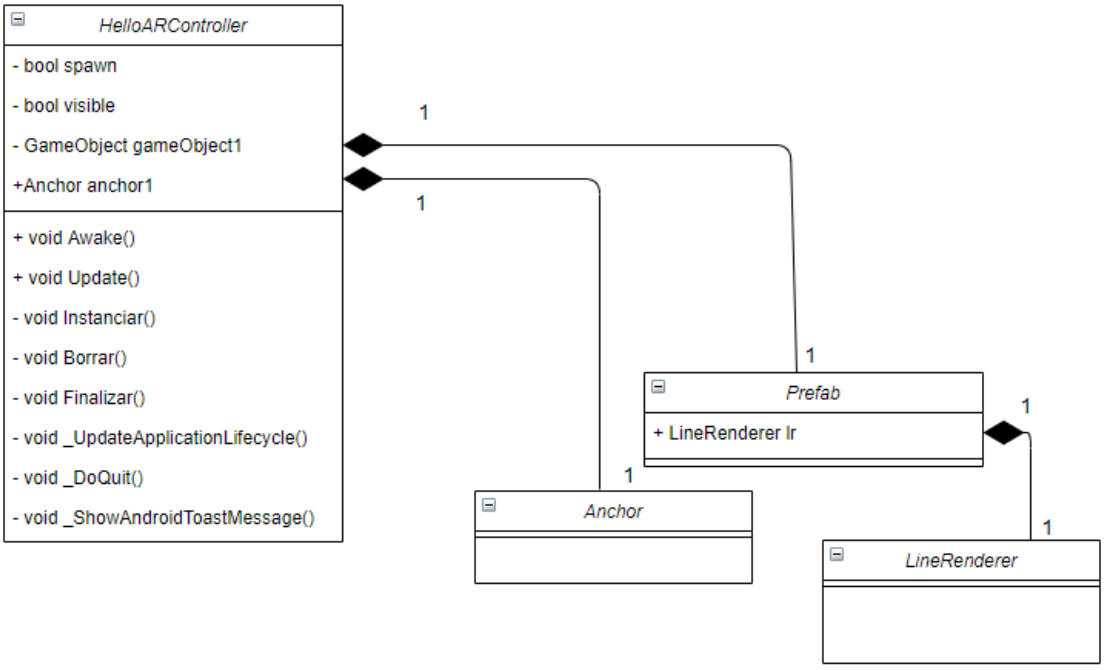

Ilustración 20 Diagrama de clases

SINGULAR BEEP

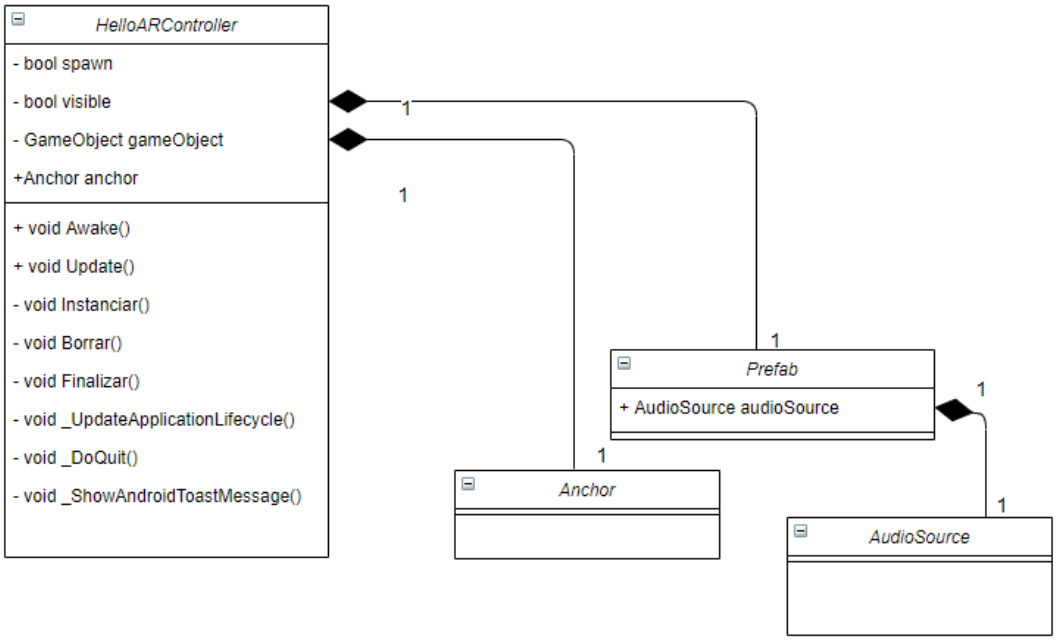

Ilustración 21 Diagrama de clases

RELATIVA DL

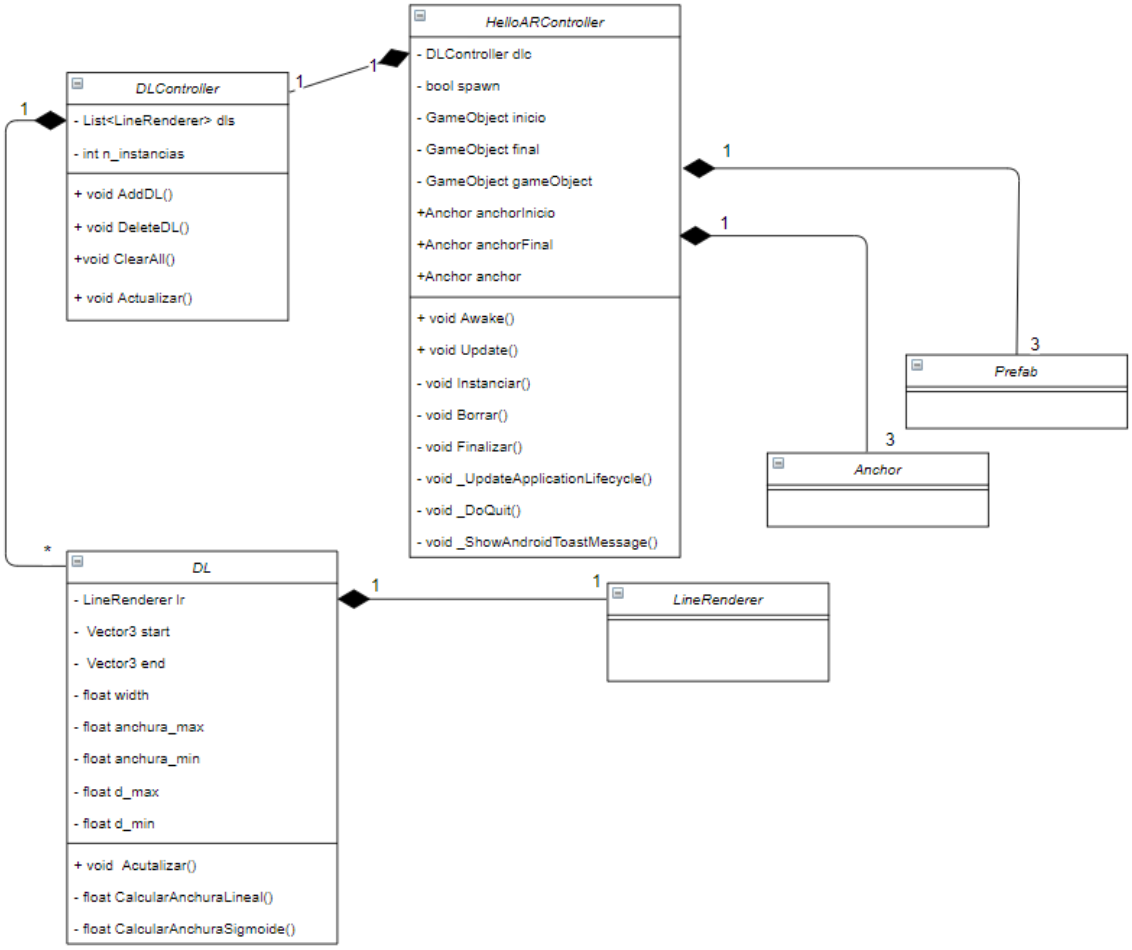

Ilustración 22 Diagrama de clases

REALTIVA EL

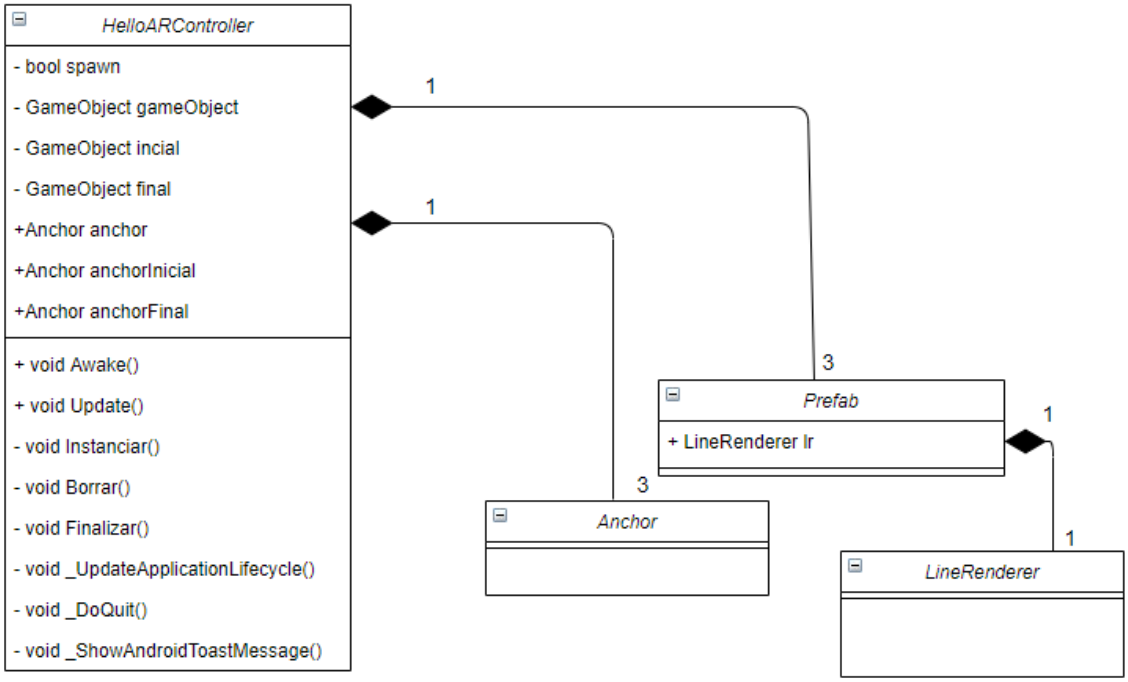

Ilustración 23 Diagrama de clases

RELATIVA BEEP

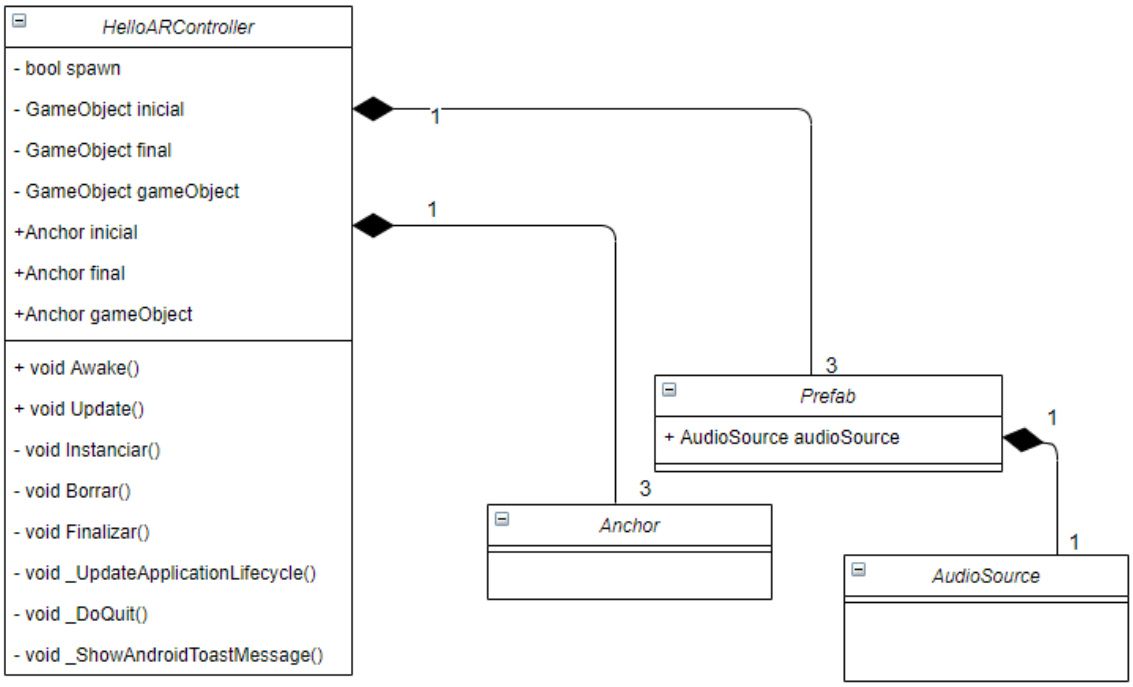

Ilustración 24 Diagrama de clases

4.7 PRUEBAS

Definiremos y explicaremos las pruebas en el apartado siguiente pero para poder explicarlas a nivel de implementación, sólo nos interesa saber que cada una consta de 10 rondas en las que se instancian 1,2 o 3 elementos en posiciones distintas.

Aunque hay diferencias entre pruebas, todas siguen la misma estructura.

4.7.1 PROCESOS PREVIOS

Antes de que comience la prueba es necesario escanear la zona de prueba, por lo que cada programa comienza con una ejecución normal de ARCore, en la cual se van detectando planos y el dispositivo los va almacenando en memoria.

Esta fase inicial finaliza cuando se toca la pantalla, lo cual da paso a la prueba. Es importante mencionar que al tocar la pantalla se calcula la colisión de la trayectoria de ese punto con el plano más cercano, esta colisión se guarda en una variable.

Una vez se ha terminado la fase inicial, se ignorarán todos los toques de la pantalla que no vayan dirigidos a elementos del interfaz.

4.7.2 TRANSCURSO DE LA PRUEBA

Como hemos mencionado, cada prueba consta de 10 rondas. Para todas las rondas a partir de la 1 es necesario borrar los objetos de la ronda anterior y crear después los nuevos. En la primera ronda no hay objetos que borrar.

Es en esta parte en la que se crean los DL y los EL asociados a los objetos, además, se almacenan los datos que estudiamos en cada prueba.

4.7.3 FINALIZAR

Después de terminar la ronda 10 se borran los objetos instanciados y los elementos de interfaz de Unity. Se ignoran todos los toques en la pantalla y aparece un mensaje que dice “Prueba finalizada”.

También se almacenan en un fichero todos los datos recogidos.

5. PRUEBAS DE USUARIO

5.1 MOTIVACIÓN

Una vez hemos diseñado los métodos de percepción de la profundidad, los debemos poner a prueba.

Para ello, debemos primero diseñar una serie de pruebas que aporten información acerca de la utilidad de los métodos en distintos ambientes. Las pruebas deben recrear funcionalidades que podríamos encontrar en aplicaciones de realidad aumentada reales.

También debemos llevar a cabo una serie de pruebas de usuario. Poder recopilar datos de distintas personas con distintos grados de familiaridad con la realidad aumentada nos dará la información que después usaremos para evaluar el desempeño de los métodos en las distintas pruebas.

5.2 MÉTODOS

Se han implementado tres métodos de percepción de la profundidad. Cada uno explora un aspecto distinto de la relación entre el objeto, entorno y el espectador. Al tratarse de maneras distintas de percibir la distancia a la que se encuentra el icono virtual, cada uno tiene una serie de debilidades y fortalezas innatas. Las pruebas están diseñadas para encontrar las diferencias entre los métodos.

A continuación, explicaremos en qué consiste cada método y las características que los diferencian del resto.

5.2.1 DISTANCE LINE (DL)

Consiste en una línea virtual que une el objeto y el usuario y cuyo grosor cambia dinámicamente según la distancia entre ambos (es más gruesa cuanto más cerca se está del objeto).

Un detalle a tener en cuenta es que, si colocásemos uno de los extremos de la línea directamente sobre la posición de la cámara, ésta taparía la visibilidad de la pantalla con su opacidad. Instanciar la punta de la línea ligeramente por debajo de la cámara resulta la solución más cómoda.

Dada la naturaleza del método el objeto debe de ser visible desde el punto de vista del espectador para no romper la inmersión, es decir, que si el objeto virtual fuese instanciado en una sala distinta o hubiese otro objeto entre ellos (como una pared) la línea, o bien atravesaría cualquier obstáculo que encontrara, o no podría funcionar al no tener la posición del objeto revelada para el sistema. Así mismo, resulta un método muy eficaz a la hora de encontrar elementos rápidamente en el espacio cercano, ya que la visibilidad de la línea hace que encontrar elementos resulte una tarea sencilla incluso para aquellos objetos que buscan mimetizarse con el entorno.

5.2.2 EARTH LINE (EL)

Crea una línea que une la base del objeto con el punto del suelo directamente debajo suyo. El grosor se mantiene constante.

La limitación más obvia de este método no afecta a aquellos objetos que estén en contacto directo con el suelo, por lo que es el único método que, dependiendo la altura del objeto, sería inservible.

Cabe mencionar que ARCore es capaz de detectar también las paredes y los techos, por lo que sería posible que los objetos se unieran con las demás superficies además del suelo. No obstante, y aunque hubiese resultado sencillo de implementar, las pruebas de usuarios se llevaron a cabo al aire libre, por lo que habría resultado un añadido completamente inútil.

5.2.3 BEEP

El principio sobre el que se basa este método es sencillo: los objetos emiten un sonido si los miras. Si apartas la vista dejan de sonar. Cuanto más lejos esté el objeto más tenue será el ruido. Cuanto más cerca, más fuerte sonará.

Este es el único método que no aporta información visual. A priori, esto permitiría recibir información acerca de los objetos sin que estos estuvieran siendo mostrados, no obstante, para aquellos casos en los que se muestran más de dos métodos en pantalla el sonido de los objetos se superpondrían, y el método dejaría de dar información útil.

Una posible solución habría sido que cada objeto emitiera un sonido distinto, sin embargo, podría darse el caso de que el usuario no supiera identificar a qué objeto pertenece qué sonido. Además, aunque el sujeto fuese capaz de distinguirlo, estaríamos condicionando su percepción acústica, ya que un sonido más grave que otro puede dar sensación distinta de lejanía.

Otra solución hubiese sido poner en pantalla un botón que permitiese al usuario acallar cada objeto. El mayor problema es que estaríamos limitando el campo visual del usuario con los elementos de la pantalla, lo cual implicaría una desventaja directa sobre las pruebas que se realizaran con los otros dos métodos.

Finalmente decidimos aplicar este método. Si bien es cierto que sacrificamos la posibilidad de aportar información espacial sin necesidad de estar mirando al objeto, resulta la opción más elegante, ya que estamos dando al usuario pleno control sin complicarlo en exceso.

5.3 PRUEBAS

A continuación, se presentan las tres pruebas realizadas por los usuarios. Para mayor claridad las llamaremos prueba *singular*, prueba *comparativa* y prueba *relativa*. Cada prueba consta de diez rondas. En cada una se instancian uno o varios objetos virtuales en distintos puntos del espacio

dependiendo de las especificaciones de la prueba. Cada una ha de realizarse tres veces, una con cada método de percepción de profundidad implementado. Para evitar que el usuario pueda ser influenciado por las respuestas que ha dado anteriormente, el orden en el que se instancia cada objeto es distinto en cada repetición.

Dada la naturaleza del software de ARCore es necesario comenzar cada prueba escaneando brevemente la zona para que el sistema de odometría del dispositivo aporte información del entorno inmediato. Esto no es estrictamente necesario, pero ayuda a evitar cargas bruscas de memoria que puedan deformar o desplazar los objetos virtuales durante la realización de la prueba.

Las pruebas comienzan cuando el usuario toca la pantalla. Este paso también es necesario, ya que el sistema calculará la colisión del vector que va desde el punto de la pantalla hasta el suelo y lo almacenará para instanciar los demás objetos virtuales.

Cuando finaliza una prueba, el sistema almacena la información recogida durante la ejecución en una carpeta del dispositivo.

5.3.1 PRUEBA SINGULAR

Objetivo: El objetivo de esta prueba es evaluar la capacidad de los métodos de aportar información precisa acerca de la posición espacial de un objeto respecto a su entorno.

Desarrollo: La prueba consta de dos partes. Durante la primera, el usuario no podrá moverse de su sitio, pero sí podrá girar la cámara y mirar en todas direcciones. A su alrededor, habrá un objeto virtual con su respectivo método de profundidad. El sujeto deberá observar dicho objeto y su entorno. Cuando considere, deberá pulsar un botón en la pantalla del dispositivo el cual dará comienzo a la segunda parte.

Al comienzo de la segunda parte el objeto virtual se hará invisible y el método de referencia usado en la prueba desaparecerá. El usuario entonces podrá moverse y deberá intentar colocar el dispositivo móvil en la posición exacta que estaba ocupando el objeto cuando era visible. Cuando el usuario considere que está en posición, podrá volver a pulsar un botón en la pantalla que instanciará el siguiente objeto y dará comienzo a la primera parte de nuevo.

Parámetros: Se medirá el tiempo que pasa cada usuario observando el objeto durante la primera parte de la prueba. También se calculará y apuntará el error entre la posición final del dispositivo y la posición real del objeto. Cabe mencionar que de esto último almacenaremos la diferencia en los tres ejes, ya que deseamos estudiar el error en cada eje por separado.

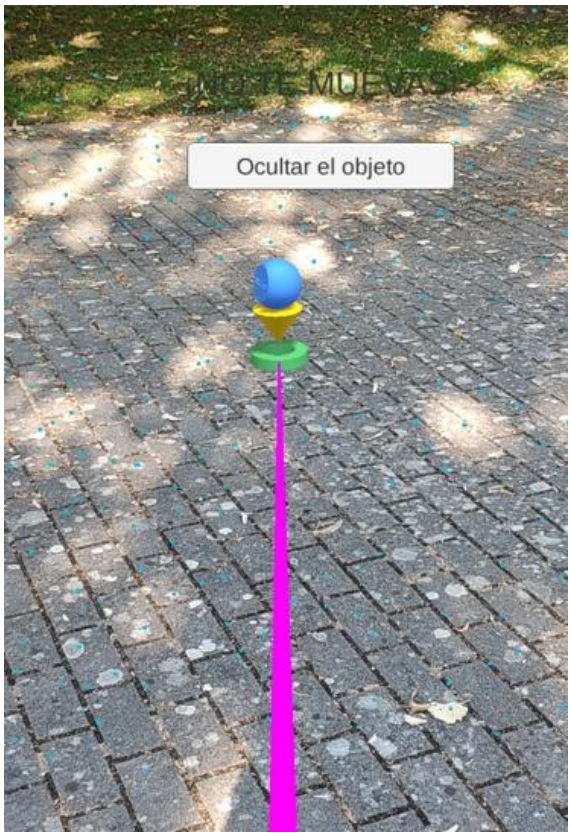

Ilustración 25 Prueba Singular con método DL y objeto visible

Ilustración 26 Prueba Singular con método DL y objeto invisible

5.3.2 PRUEBA COMPARATIVA

Objetivo: El objetivo de esta prueba es evaluar la capacidad de los métodos de representar la distancia entre un objeto y la persona que lo observa.

Desarrollo: El usuario no debe desplazarse durante la duración de la prueba. Se instancian dos objetos, uno a cada lado del sujeto de forma que no se pueden observar los dos a la vez. Hay dos botones en pantalla. El usuario debe pulsar el botón correspondiente al objeto que está situado más lejos de él. Cuando lo hace, ese par de objetos desaparece y se crean dos nuevos.

Parámetros: Se medirá el tiempo transcurrido en cada ronda, así como el número de aciertos y de fallos por parte del usuario.

Ilustración 27 Prueba comparativa con método DL

Cabe mencionar que en esta imagen el usuario debería estar colocado entre los dos objetos. Se ha tomado esta perspectiva para que quepan los dos en pantalla y se comprenda mejor la prueba.

5.3.3 PRUEBA RELATIVA

Objetivo: El objetivo de esta prueba es evaluar la capacidad de los métodos de representar la distancia entre objetos virtuales.

Desarrollo: Al comienzo se crearán dos objetos, uno inmediatamente delante del usuario (el punto de inicio) y otro bastante alejado (el punto final). Estos dos objetos permanecerán en el mismo sitio durante toda la prueba. En cada ronda se instanciará un objeto entre el de inicio y el final, formando una línea recta entre ellos. El usuario podrá desplazarse perpendicularmente a esa línea, pero no podrá ponerse en paralelo a ella. En la pantalla del dispositivo habrá un slider. El objetivo de la prueba es colocar el slider de forma que imite la posición relativa del objeto del medio respecto al de inicio y el final.

Parámetros: Se medirá el tiempo transcurrido en cada ronda, así como el error de la respuesta.

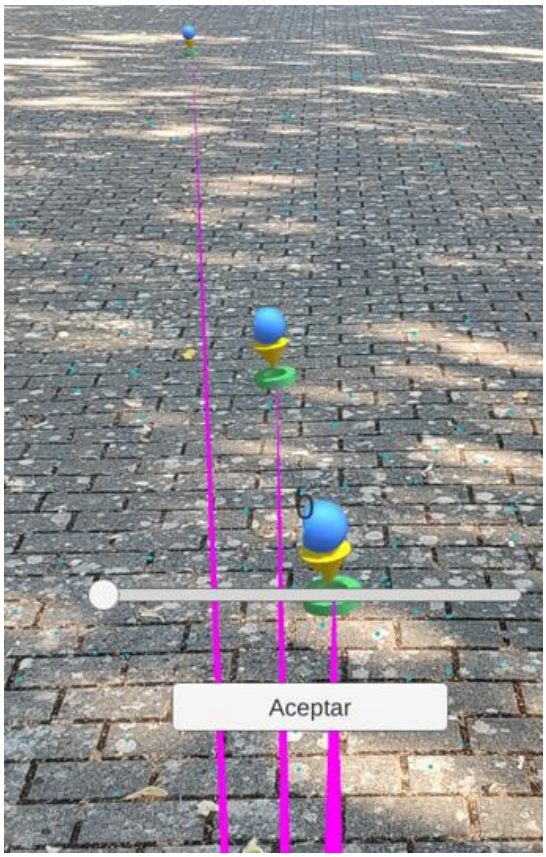

Ilustración 28 Prueba relativa con método DL

5.4 DESARROLLO DE LAS PRUEBAS DE USUARIO

Hay tres pruebas que se deben realizar con cada uno de los tres métodos. Esto hace un total de 9 pruebas. Como queremos evaluar los métodos por separado, los usuarios realizarán tres pruebas seguidas con el mismo método y después realizarán un cuestionario.

Previsiblemente, el primer método que se utilice será el que peores resultados aporte, ya que el usuario no estará todavía familiarizado con la tecnología ni con el desarrollo de las pruebas, por ello, vamos a turnar el orden en el que se realizan. Hay seis permutaciones en total:

1. DL-El-Beep
2. DL-Beep-EL
3. EL-DL-Beep
4. EL-Beep-DL
5. Beep-DL-EL
6. Beep-EL-DL

El cuestionario a realizar después de cada grupo de tres pruebas es el mismo para los tres métodos. Hemos decidido utilizar el NASA-TLX y el System Usability Scale por su eficacia testada. Al final de las preguntas las preguntas de esos dos cuestionarios, preguntamos al usuario que ordene las pruebas de acuerdo a lo útil que le ha resultado el método a la hora de realizarla.

Además, hay otro cuestionario al comienzo de la prueba de usuario donde preguntamos el nombre, la edad, el género y el nivel de experiencia con sistemas de realidad aumentada o similares.

5.5 RESULTADOS

Se han realizado un total de 8 pruebas de usuario a 5 hombres y 3 mujeres con una media de edad de 27 años y, en su mayoría, un nivel básico de experiencia de manejo de sistemas de realidad aumentada.

Las pruebas de usuario se han realizado a lo largo de una semana. Originalmente se quería evaluar a 12 usuarios. Por problemas de tiempo, ese número se ha visto reducido a 8.

Es importante mencionar que en dos ocasiones, durante el transcurso de la prueba Singular-Beep, el sistema falló y los datos de dichas pruebas no se guardaron correctamente. Hemos dejado ambas pruebas fuera de las estadísticas para evitar que aportaran datos erróneos.

A continuación, presentaremos y explicaremos los datos recogidos prueba a prueba.

5.5.1 RESULTADO PRUEBA COMPARATIVA

Los dos parámetros observados durante la prueba son la duración (en segundos) y la media del número de aciertos.

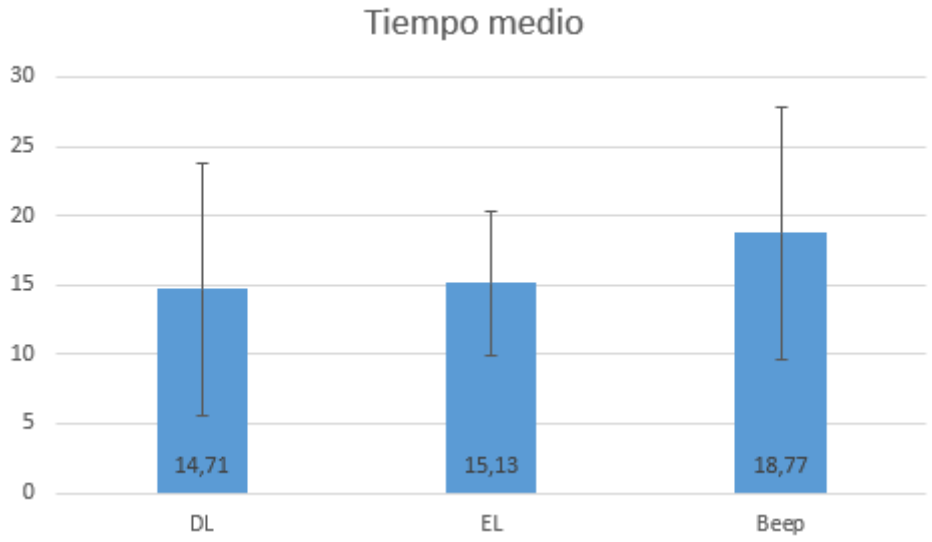

Ilustración 29 Tiempo medio de la prueba comparativa

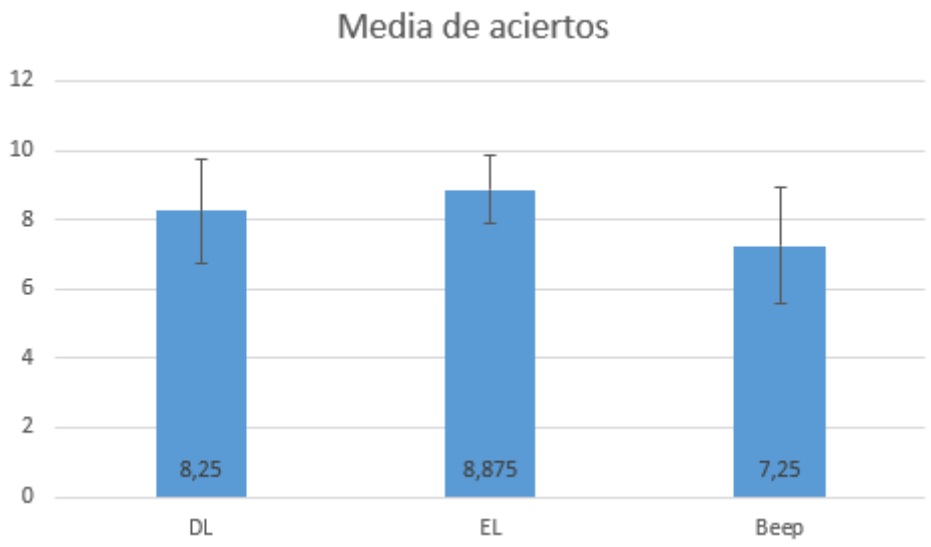

Ilustración 30 Media de aciertos de la prueba comparativa

El DL y el EL han dado resultados bastante parecidos, siendo el EL ligeramente superior.

Tiene sentido que el DL haya dado buenos resultados, ya que aporta información visual acerca de la distancia a la que se encuentran los objetos. La mayoría de los usuarios ha comparado la anchura de las líneas para determinar qué objeto está más lejano.

No obstante, cabría esperar que el Beep hubiese dado mejores resultados, ya que la táctica a seguir en esta prueba parecía sencilla de ejecutar: escuchar el objeto que emite el sonido más tenue y elegirlo. Esto parece indicar que la información sonora es más difícil de procesar para los usuarios.

En cuanto a los tiempos de desarrollo de la prueba, son todos bastante similares, aunque los peores resultados los da el Beep. Esto puede indicar que es más rápido reconocer y procesar la información adquirida visualmente que la sonora.

5.5.2 RESULTADO PRUEBA SINGULAR

Los parámetros observados durante la prueba son la duración (en segundos), el error en los ejes X-Z y el error del eje Y. Hemos elegido mostrar la diferencia entre los ejes de esta forma porque los ejes X-Z responden a la diferencia en el plano horizontal mientras que el plano Y es la diferencia de altura. Queremos observar la forma en la que los usuarios son capaces de entender la distancia y la altura por separado.

La desviación no está en metros, sino en la unidad de referencia de Unity.

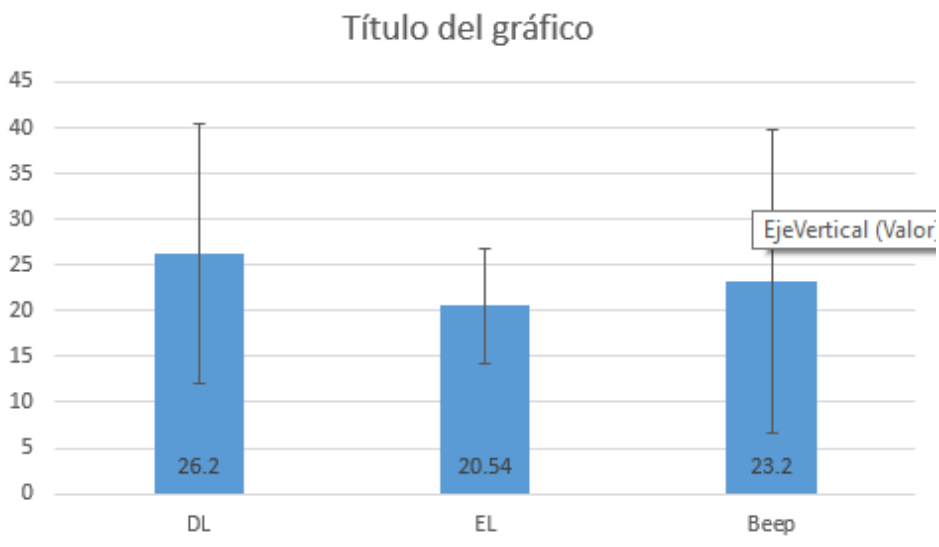

Ilustración 31 Tiempo medio de la prueba singular

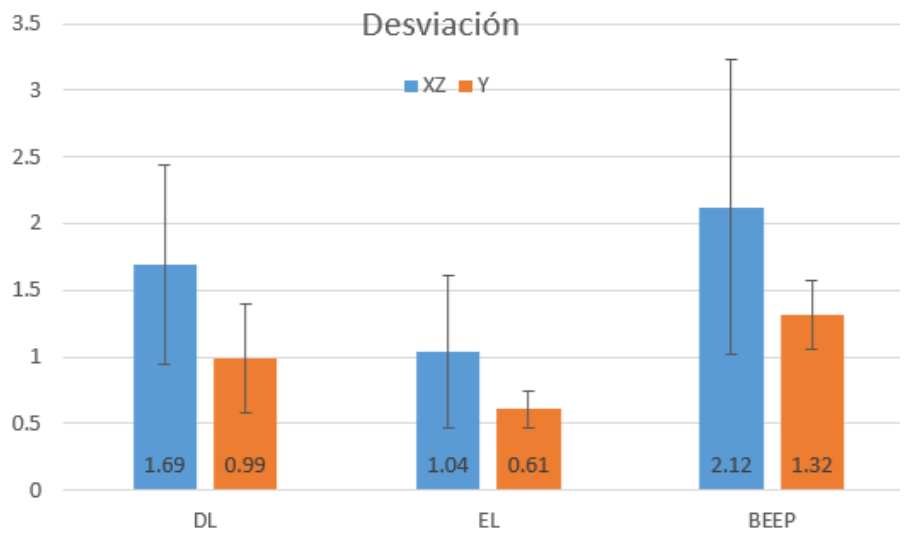

Ilustración 32 Media de aciertos de la prueba singular

Los resultados de esta prueba son previsibles. Por lo general, los usuarios han sido más eficaces a la hora de interpretar la altura que la distancia. Además, el EL ha resultado ser el método más eficaz.

A la hora de realizar la prueba con el método EL, muchos usuarios tomaban como referencia el punto en el que la línea tocaba el suelo y se desplazaban mirando ese punto en concreto hasta estar colocados sobre él. Esto cambiaba con los otros dos métodos, en ambos casos, la mayoría de los usuarios se desplazaban mirando a la pantalla. Esto indica que el método EL es más eficaz a la hora de dar puntos de referencias físicos.

También se puede observar que el método Beep ha sido más rápido y ha dado peores resultados que el DL. Al no aportar información precisa en esta prueba, la mayoría de los usuarios utilizaban el método únicamente para localizar el objeto, y una vez localizado, se guiaban solo por los elementos visuales. En contraste, los usuarios sí que observaban más detenidamente la línea que les unía al objeto con el DL, y estudiaban más a fondo la información que les proporcionaba.

5.5.3 RESULTADO PRUEBA RELATIVA

Los parámetros observados durante la prueba son la duración (en segundos) y el error cuadrático medio en las respuestas.

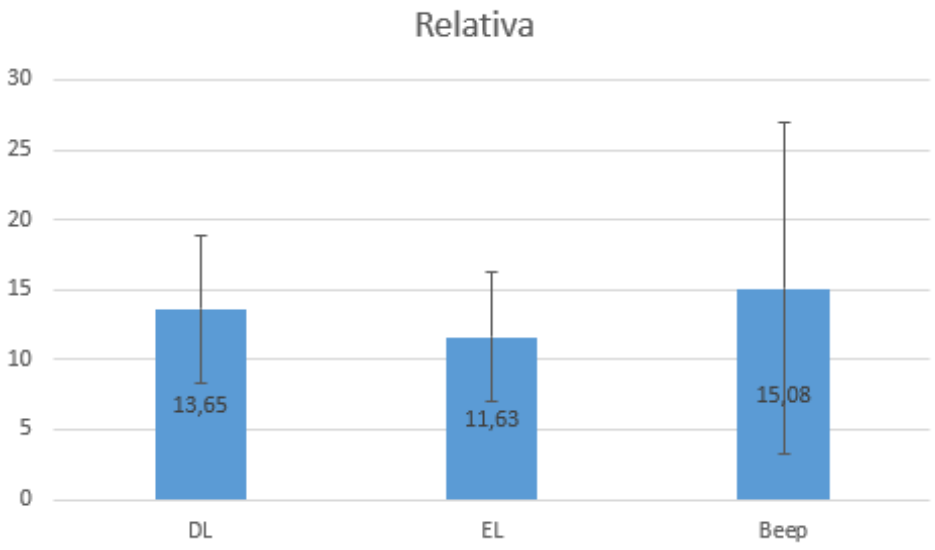

Ilustración 33 Tiempo medio de la prueba relativa

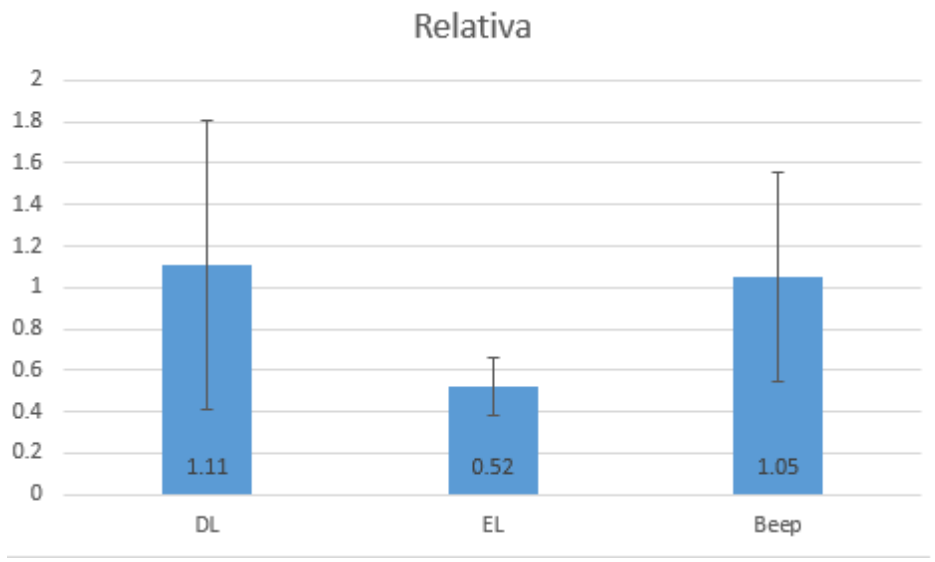

Ilustración 34 Error medio de la prueba relativa

Esta es la única prueba en la que el método DL ha dado los peores resultados. Quizá se deba a que, al haber 3 objetos, las líneas que salían de ellos ocupaban una parte mayor de la pantalla y empeoraban la visibilidad de la prueba.

De nuevo, el método Beep es el que más tiempo necesita para terminar la prueba. Quizás se deba a que, con 3 objetos instanciados, es más complicado ser selectivo a la hora de mostrar uno u otro, y los usuarios se habrán tenido que ver obligados a desplazarse o a poner la cámara ángulos extraños para poder escuchar cada sonido por separado.

5.6 CONCLUSIONES

El mejor método en todas las pruebas ha resultado ser el EL. Por lo general, a la hora de realizar las pruebas los usuarios tomaban como referencia el punto en el que la línea virtual tocaba el suelo y sacaban sus propias conclusiones del mundo físico. De acuerdo con los formularios, también resulta el método más sencillo de entender y utilizar.

El método Beep ha sido el peor en términos generales. La información auditiva que aportaba ha sido la más lenta en ser procesada por los usuarios. Otro factor a tener en cuenta es el ritmo al que decae la potencia del sonido que emitían los objetos no es lineal, por lo que la información que aporta no es la más válida a la hora de hacer cálculos precisos. También es el método más impopular entre los usuarios.

Finalmente, el método DL ha dado unos resultados bastantes similares al EL, aunque ligeramente inferiores. La única desventaja que parece presentar respecto a los otros dos métodos es que puede llegar a tapar información relevante en pantalla. Los usuarios consideran que este método es el que más esfuerzo mental requiere para poder ser usado.

Aunque haya dado malos resultados, resultaría interesante seguir investigando el sonido como una forma de aportar información. Ahora que sabemos que no es capaz de aportar información precisa, podríamos diseñar casos de uso concretos en los que se premie la interacción que no dependa del contacto visual, de esta forma, podríamos explorar el verdadero potencial detrás de la idea. El sonido que emite también parece ser un parámetro importante a tener en cuenta, ya que un sonido estridente como el pitido que hemos usado parece tener efectos negativos sobre la capacidad de interpretación de los usuarios.

La mayor fortaleza del método DL es la facilidad con la que permite encontrar los objetos a los que está unido. También es cierto que eso limita sus usos, ya que es un elemento que rompe la inmersión y reduce el campo visual, pero posee un gran potencial en aplicaciones en las que importe localizar elementos tanto reales como virtuales.

Finalmente, el mejor método a la hora de mostrar la distancia era aquel que permitía al usuario poner los objetos virtuales en función de referencias físicas. Esto puede significar un nuevo tipo de relación entre objetos reales y virtuales. Anteriormente hemos hablado de cómo se intenta activamente que los objetos virtuales se comporten como objetos reales con el objetivo de aportar información espacial, no obstante, puede resultar igual de efectivo centrar nuestros esfuerzos en mostrar la distancia del objeto virtual a otros objetos reales de su entorno.

6.ANEXO

6.1 FORMULARIOS

6.1.1 FORMULARIO INCIAL

Formulario inicial

1 . Nombre

_____

2. Género

_____

3. Edad

_____

4. ¿Qué nivel de experiencia tiene usted con la realidad aumentada o con otras tecnologías similares?

Marca solo un óvalo.

- Ninguno
- Básico
- Intermedio
- Avanzado
- Experto

e

6.1.2 FORMULARIO DE LAS PRUEBAS

1. Escriba el orden en el que ha realizado esta prueba

Marca solo un óvalo.

1	2	3
◯	◯	◯

Responda a las siguientes preguntas asignando números del 1 al 5:
1.Muy en desacuerdo, 5-Muy de acuerdo

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

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo	◯	◯	◯	◯	◯	Muy de acuerdo

3. 2. Encuentro este sistema innecesariamente complejo.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo	◯	◯	◯	◯	◯	Muy de acuerdo

4. 3. Creo que este sistema es fácil de utilizar.

Marca solo un óvalo.

e

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

5. 4. Creo que necesitaría la ayuda de un técnico para poder utilizar este sistema.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

6. 5. Encuentro las diferentes funciones de este sistema bien integradas.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

7. 6. Creo que hay demasiadas inconsistencias en el sistema.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

8. 7. Creo que la mayoría de la gent aprendería a utilizar este sistema muy rápido.

Marca solo un óvalo.

e

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

9. 8. Creo que el sistema es incómodo de utilizar.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

10. 9. Me he sentido muy seguro utilizando este sistema.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

11. 10. He tenido que aprender muchas cosas antes de poder ponerme en marcha con el sistema.

Marca solo un óvalo.

	1	2	3	4	5	
Muy en desacuerdo						Muy de acuerdo

Responde las siguientes preguntas asignando números del 1 al 10.

1. Muy bajo
10.Muy alto

2 1. ¿Cómo de mentalmente exig nte ha sido la prueba?

Marca solo un óvalo.

	1	2	3	4	5	6	7	8	9	10	
Muy baja	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muy alta

13. 2. ¿Cómo de físicamente exigente ha sido la prueba?

Marca solo un óvalo.

	1	2	3	4	5	6	7	8	9	10	
Muy baja	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muy alta

14. 3. ¿Cómo de rápido ha sido el ritmo de realización de la prueba?

Marca solo un óvalo.

	1	2	3	4	5	6	7	8	9	10	
Muy baja	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muy alta

15. 4. ¿Cómo de exitoso has sido a la hora de lograr los objetivos de la prueba?

Marca solo un óvalo.

	1	2	3	4	5	6	7	8	9	10	
Muy baja	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muy alta

6 5. ¿Cómo de duro has tenido qu trabajar para alcanzar ese nivel de actuación?

Marca solo un óvalo.

	1	2	3	4	5	6	7	8	9	10	
Muy baja	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muy alta

17. 6. ¿Cómo de inseguro, desalentado, irritado, estresado y molesto te encontrabas?

Marca solo un óvalo.

	1	2	3	4	5	6	7	8	9	10	
Muy baja	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Muy alta

Responda a la siguiente preguntad de forma libre

18. ¿Cómo de útil le ha resultado el sistema a la hora de llevar a cabo las pruebas? Ordénelas de mayor a menor utilidad.

Marca solo un óvalo por fila.

	1 (Más útil)	2	3 (Menos útil)
Comparativa	◯	◯	◯
Singular	◯	◯	◯
Relativa	◯	◯	◯

9 A continuación, escriba cualquier observación que le haya tenido y que le parezca relevante.

6.2 CÓDIGO DE JUPYTER NOTEBOOK

Este es el código que se ha utilizado para extraer y editar los datos recopilados de las pruebas.

6.2.1 LIBRERÍAS

Hemos utilizado las siguientes librerías:

- Sklearn, librería científica muy potente. Nosotros vamos a usar su funcionalidad para sacar los errores de conjuntos de datos.
- Panda, librería que permite tratar con tablas de datos.
- Listdir, permite recorrer directorios y archivos.

6.2.2 PREPROCESAMIENTO DE LOS DATOS

Los datos se almacenan en formatos distintos dependiendo de la prueba, por lo que es importante reescribirlos a un formato csv separado por comas que haga más sencillo su tratamiento. Vamos a reescribir los datos de forma que estén ordenados por columnas, que es el formato que panda necesita para poder leer todos los datos desde un fichero de texto.

Los datos de las pruebas comparativas se almacenan de la siguiente forma:

```
B A B B B A A A A B
39.66
21.89999
7.340004
11.92
13.06
2.800003
2.720001
21.98
6.699997
```

Donde la primera fila son las respuestas que el usuario ha dado y el resto son los segundos que ha tardado en cada ronda. En las respuestas, la “A” corresponde a haber elegido el objeto de la izquierda como el más lejano y la “B”, al de la derecha.

```
def leerComparativa(path):
    f = open(path)
    linea = f.readline()
    linea.split(" ")
    respuestas = [x for x in linea if (x == "A") | (x == "B")]
```

Lo primero que hacemos es abrir el fichero y leer la primera línea para almacenar los datos en una lista de Python.

```
tiempos = []
linea = f.readline()
while linea:
    tiempos.append(linea[:-2])
    linea = f.readline()
```

Al leer los tiempos hay que tener en cuenta que se lee la línea entera, incluyendo el salto de línea, por lo que tenemos que dejar sin escribir los dos últimos caracteres (que corresponden al \n).

Los datos de las pruebas singular tienen el siguiente formato:

```
(0.0, -0.1, -0.5)    ---->  62.59999
(0.1, 0.0, 0.2)    ---->  38.62
(-1.2, 0.1, 2.0)    ---->  36.62001
(0.4, 0.0, 0.0)    ---->  53.60001
(0.0, 0.2, -0.5)    ---->  32.56
(-0.3, 0.2, 0.8)    ---->  63.60001
(1.3, -0.3, 0.5)    ---->  21.34
(-0.1, 0.0, -0.2)    ---->  30
(0.2, 0.0, 0.1)    ---->  24.44
(1.1, 0.1, 0.9)    ---->  28.56
```

Los primeros números son los errores en los tres ejes. Tienen ese formato porque es como Unity almacena las posiciones. El número de la derecha es el tiempo.

```
x = []
y = []
z = []
tiempos = []

f = open(path)

linea = f.readline()
while linea:
    partes = linea.split("  ")
    aux = partes[0][1:-1].split(",")
    x.append(aux[0])
    y.append(aux[1])
    z.append(aux[2])
    tiempos.append(partes[2][:-2])
    linea = f.readline()
```

Lo que hacemos es leer una línea entera y partirla por los espacios en blanco, de esta forma nos queda dividida en tres trozos: los vectores, la flecha y el tiempo. El tiempo lo podemos añadir eliminando los dos últimos caracteres de salto de línea. Para poder sacar los errores en cada eje primero eliminamos los paréntesis del principio y del final del string, después lo dividimos por las comas, y finalmente, metemos cada eje en su correspondiente lista.

Los datos de las pruebas relativas se almacenan de la siguiente forma:

```
0.2012124 0.8037491 0.4960125 0.4052697 0.6114138 0.3470759 0.4023107 0.9526858 0.9526858 0.8123023
42.14
7.879997
4.440002
17.72
9.539993
9.480003
6.32
3.339996
1.479996
12.36
```

La primera línea son los errores realizados. El resto de las líneas son los tiempos. Este formato se lee de la misma manera que el de las pruebas comparativas.

6.2.3 CÁLCULO DE ERRORES

Para calcular los errores de las pruebas comparativas, tenemos que tener primero las soluciones correctas. Como cada prueba tenía una permutación distinta, tenemos que guardar las tres posibles ristras de soluciones. Después, leemos las respuestas que ha dado el usuario y comparamos ambas listas. El resultado es un array de booleanos con 1`s en las respuestas correctas y 0`s en las incorrectas. Si sumamos ese array, tenemos una puntuación sobre 10.

Para la prueba singular, tras leer los datos calculamos el error cuadrático medio en los ejes X-Z y el del eje Y por separado.

En el caso de la prueba relativa, calculamos el error cuadrático medio y lo mostramos. Unity nos devuelve los errores en base 1, nosotros vamos a multiplicar los resultados sobre 100 para evaluar de manera más cómoda los resultados.

A la hora de recorrer los ficheros, tenemos que tener en cuenta si el nombre contiene el nombre de la prueba y del método que estamos evaluando. Si ambos coinciden, leemos los datos y comparamos los resultados para mostrarlos por pantalla.

```
def acierto_comparativa():

    respuestas = {"DL" : ["A","B","B","A","B","A","B","A","A","A"],
                  "EL" : ["B","A","B","B","A","A","A","A","A","B"],
                  "Beep" : ["A","B","A","B","A","B","B","A","A","A"]}

    path = "DatosProcesados/Comparativo/"

    for metodo in ["DL","EL","Beep"]:
        print("--------------------------")
        print(metodo)
        print("--------------------------")
        for file in listdir(path):
            if (metodo in file):
                comp = pd.read_csv(path+file)
                print(file + ": " + str(np.sum(comp["respuestas"] == respuestas[metodo])))
```

```
def error_singular():

    path = "DatosProcesados/Singular/"

    for metodo in ["DL","EL","Beep"]:
        print("--------------------------")
        print(metodo)
        print("--------------------------")
        for file in listdir(path):
            if (metodo in file):
                comp = pd.read_csv(path+file)
                errores = list(round(np.sqrt(comp["x"]**2 + comp["z"]**2),2))
                print(file + ": " + str(errores))
                print("------->" + str(list(comp["y"])))
```

```
def error_relativa():

    respuestas = {"DL" : [0.15,0.75,0.50,0.45,0.6,0.25,0.35,0.9,0.8,0.7],
             "EL" : [0.6,0.8,0.75,0.5,0.25,0.7,0.15,0.45,0.9,0.35],
             "Beep" : [0.7,0.5,0.9,0.6,0.8,0.35,0.75,0.45,0.15,0.25]}

    path = "DatosProcesados/Relativo/"
    for metodo in ["DL","EL","Beep"]:
        print("--------------------------")
        print(metodo)
        print("--------------------------")
        for file in listdir(path):
            if (metodo in file):
                comp = pd.read_csv(path+file)
                errores = list(round((comp["respuestas"] - respuestas[metodo]) * 100,0))
                media = mean_squared_error(comp["respuestas"] , respuestas[metodo])*100
                print(file + ". Media: "+ str(media) + " ----> " +str(errores))
```

6.2.4 CÁLCULO DE LA MEDIA DEL TIEMPO

Como los datos de los tiempos de realización son iguales en todos los casos, solo necesitamos un único script para todas las pruebas.

```
def tiempos(prueba):

    path = "DatosProcesados/" + prueba + "/"

    tiempos = {"DL" : [], "EL" : [], "Beep":[]}

    for metodo in ["DL","EL","Beep"]:
        for file in listdir(path):
            if (metodo in file):
                comp = pd.read_csv(path+file)
                if (len(comp["tiempos"]) != 0):
                    fin = 9
                    if (comp["tiempos"][fin] == 0):
                        fin = 8
                    tiempos[metodo].append(round(sum(comp["tiempos"][1:fin])/len(comp["tiempos"][1:fin]),2))

        print(metodo + ": " + str(round(sum(tiempos[metodo])/len(tiempos[metodo]),2)))
    print("---------------")
```

Lo que hace este programa es recorrer los ficheros buscan coincidencias con el tipo de prueba y el tipo de método como hemos hecho antes. Leerlos, y sacar la media por pantalla.

6.3 DATOS

A continuación vamos a mostrar todos los datos recogidos durante las pruebas de usuario. Vamos a mostrar los errores por un lado y los tiempos por otro.

6.3.2 PRUEBA COMPARATIVA

Número de aciertos:

DL	EL	Beep
10	10	10
6	9	5
8	8	6
9	9	7
7	9	6
10	10	9
9	9	7
7	7	8

Tiempos:

	64,98	6,120003	11,22	10,87999	5,76001	11,08	9,159988	7,320007	5
	97,84	25,95999	64,98	32,75999	29,12	18,36002	5,76001	62,53998	15,34
Beep	29,56	7,98	11,76	34,3	7,480003	14,98	7,800003	6	8,259995
	25,3	0,05999756	19,14	14,06	5,18	4,839996	7,660004	2,659996	7,040001
	34,28	7,900002	44,98	14,78	19,3	9,639999	5,720001	19,98	8,619995
	26,66	7,759995	26,48	17,5	3,399994	13,04	3,699997	11,3	16,74001
	80,53999	0,3000031	21,77999	22,46001	5,880005	12,53999	10,08	35,95999	16,5
	67,44	5,5	10,3	26,3	9,519989	15,60001	6,539993	2,980011	11,34
	46	10,38	11,3	4,82	20,56001	8,339996	4,279999	3,619995	3,880005
	25,64	5,939999	12,2	5,880005	5,459999	3,879997	6,660004	39,69999	26
EL	26,56	14,56	16,28	26,62	12,48	14,58	11,53999	29,84	2,880005
	35,2	11,74	7,799995	13,94	3,059998	2,980003	3,540001	14,12	6,139999
	44,5	6,339996	10,76001	9,219986	16,60001	8,559998	10,44	11,84	3,580002
	39,34	11,08	7,919998	4,620003	4,239998	10,18	7,159996	8,900002	5,020004
	39,66	21,89999	7,340004	11,92	13,06	2,800003	2,720001	21,98	6,699997
	49,52	32,75999	5,68	5,880005	39,06	49,38	35,53999	17,44	7,5
	14,4	2,400002	4,419998	13,72	6,440002	4,259995	4,900002	8,980003	6,519997
	26,46	4,639999	11,9	33,22	29,14	1,360001	6,279999	36,56	25,56
DL	8,52	4,019999	12,38	31,56	8,720001	12,52	34,06	6,380005	7,23999
	18,9	4,380001	4,759998	31,66	4,82	6,899994	6,800003	3,799995	9,100006
	13,26	7,84	5,239998	16,84	12,10001	22,3	27,8	6,380005	9,979996
	20,38	5,840004	6,5	11,46	10,98	10,88	8,579994	12,36	10,58
	46,7	12,96	9,779999	11,26	23,26	6,5	47,37999	13,70001	6,979996
	28,38	6,639999	26,44	27,02	30,66	31,48	26,42001	6,299988	20,22

6.3.3 PRUEBA SINGULAR

Errores en los ejes X-Z:

	0,76	0,1	2,36	1,39	2,47	1,68	1,84	0,36	0,45	5,02
	0,51	0,81	2,02	3,83	7,62	4,08	0,3	0,45	0,61	2,91
	1,66	1,2	3,48	6,42	1,14	0,4	1,02	1,02	4,52	4,75
DL	0,54	0,1	2,2	3,71	3,06	0,81	0,61	0,22	0,76	0,72
	0,86	0,45	0,14	1,4	0,51	2,71	0,67	3,61	0,82	4,08
	0,4	0,1	0,45	0,7	2,02	0,81	0,71	0,4	0,32	1,56
	0,5	0,22	2,33	0,4	0,5	0,85	1,39	0,22	0,22	1,42
	0,36	0,45	0,73	4,13	3,61	2,91	5,46	3,42	0,5	4,95
	0,71	1,12	1,53	1	0,2	0,22	0,14	0,57	0,1	0,82
	0,41	0,32	1,44	2,33	0,45	0,85	0,22	0,42	0,22	1,14
	0,41	1,73	1,5	0,89	1,1	0,22	0,1	0,2	4,06	0,86
EL	0,41	1,42	0	0,71	2,06	0,1	0,28	0,72	0,14	0,28
	1,02	5,9	7,42	3,4	1,02	0,51	0,14	1,84	1,66	0,61
	0,7	0,32	2,02	0,32	0,22	2,06	0,42	1,03	0,1	0,2
	0,3	0,78	0,94	1,24	0,5	0,36	0,64	1,5	1,26	0,92
	0,3	1,56	3,33	2,94	1,73	0,14	0,5	0,78	0,63	0,71
	3,11	0,63	0,28	2,64	3	0	4,72	2,88	3,13	0,95
	4,42	0,64	0,32	6,12	3,32	5,75	0,94	3,14	4,7	1,24
	0,22	0,45	0,92	0,7	3,3	0,51	0,57	4,25	1,08	0,4
Beep	0,32	2,79	3,13	9,32	12,32	0,41	3,8	2,25	2,42	0,63
	0,28	0,45	0,61	0,42	1,79	0,36	1,49	1,56	0,7	0,6
	0,3	1,1	0,71	2,59	2,11	0,45	3,33	2,88	2,9	0,82

Errores en el eje Y:

	-0,2	0,2	0,1	0	0,3	0,1	0,1	0,4	0,2	0,2
	0,6	0,3	-0,1	0,2	0,2	0,2	0,2	-0,2	0,1	0,2
	-0,2	0,6	0,3	0	0,4	0,3	0,5	-0,1	0,2	-0,1
DL	0,2	0,3	0,5	-0,4	-0,3	1,3	0,4	0,8	-0,1	-0,4
	-0,4	0,3	0,2	-0,1	0	0,1	0,2	-0,2	0	1,1
	0	0,1	0,4	0,1	0,2	0,5	0,4	0	0,1	-0,3
	-0,1	0	0,1	0	0,2	0,2	-0,3	0	0	0,1
	0,3	0,1	-0,1	0,2	-0,1	0,3	0,5	0,3	0,1	0,6
	0,1	0,2	0,4	0,1	0	0,2	0,1	0	0	0,3
	0	0,2	0,1	0,2	-0,2	0,1	0,2	-0,2	0,3	-0,3
	0	0,2	0,2	0,5	0	0,2	0,5	0,1	-0,1	0,3
EL	0	0,2	0,1	0,2	0,3	0,2	0,2	-0,1	0,2	0,1
	0,2	0,2	0,4	0,4	0	0,3	0	-0,2	0,2	0,1
	0,2	0,1	0,1	0,2	-0,2	-0,3	0,2	0	0	0,1
	-0,1	0	0,1	0,3	0,1	0,1	0,3	-0,1	0	0
	0	-0,1	-0,1	-0,2	-0,1	0,1	-0,1	0,1	0,1	-0,3
	0,1	0,2	0,1	0	-0,6	0,1	-0,1	-0,2	-0,2	0,2
	0,5	-0,5	0,1	0	0,4	0,2	0,1	0	0,2	0,1
Beep	0,2	-0,1	-0,1	0	0,3	0,1	0,2	-0,2	0	0,5
	0,2	0,1	0,5	0,4	0,7	0,5	0,3	0,2	0,2	0,5
	0,4	0,1	0	-0,1	0	0,2	0	-0,1	0,1	-0,1
	0	-0,2	0,4	0,2	0,5	0,1	0,2	0,1	-0,1	0

Tiempos:

	13,08	6,720001	6,639999	6,859993	9,040001	5,939995	3,280014	5,779999	8,059998	8,459991
	28,77999	70,74001	40,72	81,98	55	67,66	52,80002	50,16003	48,94	34,01996
	18,5	30	17,53999	18,06	22,42001	14,08	21,39999	19,80002	24,48001	20,14001
	44,22	53	34,86	17,02	16,54001	12,56	11,22	13	8,279999	16,07999
DL	38,8	15,34	17,36	28,48	11,86	17,5	8,23999	8,61998	9,899994	28,36002
	24,64	25,27999	10,36	17,32	32,44	16,84	20,92	16,3	12,02	14,74001
	62,59999	38,62	36,62001	53,60001	32,56	63,60001	21,34	30	24,44	28,56
	33,1	23,24	28,75999	27,34	24,02	30,60001	39,39999	27,48001	22,76001	34,36002
	31,56	7,299995	5,940002	10,48	34,24	8,659988	5	12,78	9,199997	8,140015
	24,58	43,68	43,26	159,48	29,62	131,18	72,81998	18,30005	17,82001	17,20001
	18,5	7,620003	11,84	11,9	22,68	11,72	5,139999	12,94	13,88	6,919998
Beep	16,68	3,879997	7,620003	11,04	51	11,8	3,540009	11,88	8,039993	8,080017
	28,52	17,7	14,78001	15,66	42,79999	24,68001	4,779999	18,88	9,440002	19,39999
	22,84	16,96	55,68001	16,46001	21,34	20,65999	10,08	18,86	21,37997	42,85999
	26,7	12,88	10,84	7,800003	5,860001	7,580002	19,8	5,76001	9,259995	5,62001
	21,38	18,3	8,900002	14,02	35,64	8,559998	82,21999	9,099976	18,88	20,80002
	30,48	17,46	10,74001	12,62	18,2	15,94	24,45999	11,12	15,56	10,8
EL	18,32	20,04	17,18	16,84	33,16	19,48001	17,94	13,16	27,51999	10,30002
	18,68	19	12,58	9,339996	10,25999	8,519989	12,72	24,42	8,459991	11,16
	26,62	22,26	33,32	9,279999	69,86	27,84	26,78	15,16	17,98001	10,48001
	44,66	15,92	40,44	44,26001	31,72	23,15999	59	6,100006	17,12	8,639984
	29,4	44,32	26,64	19,92	49,16	11,34	15,91998	21,91998	10,39999	17,5

6.3.4 PRUEBA RELATIVA

Errores:

	9	5	10	9	14	10	14	4	10	14
	-4	6	14	14	16	-7	-6	4	9	15
	11	8	18	23	16	14	15	1	6	10
DL	10	5	10	10	10	15	15	5	10	10
	3	8	7	7	3	5	-1	5	8	9
	17	13	19	18	20	21	16	3	8	10
	5	5	0	-4	1	10	5	5	15	11
	4	1	14	9	5	5	4	2	6	2
	4	-3	-5	8	16	4	5	6	-2	13
	12	2	2	2	-14	6	-9	-3	-5	-1
	4	-6	-1	0	11	9	10	7	7	13
EL	5	0	5	10	11	5	0	10	5	15
	10	1	1	0	-4	5	2	12	5	10
	-1	2	-4	4	13	1	5	10	1	16
	0	-2	2	-2	5	-5	12	5	1	7
	10	4	2	6	9	6	3	7	6	9
	3	-2	1	9	8	-6	3	7	-1	4
	-1	7	2	3	8	8	17	16	19	15
	5	13	2	9	-2	15	1	7	12	11
Beep	0	10	0	15	5	15	5	16	15	19
	-1	-6	-11	-10	-12	-1	-8	5	-4	-5
	5	25	0	6	2	11	5	9	10	16
	0	11	-9	8	-4	14	5	9	13	8
	-2	8	-9	9	-11	-4	20	22	21	14

Tiempos:

	23,08	6,720001	4,5	4,379997	3,260002	3,699997	13,46	29,78001	12,37999	3,400009
	97,53999	23,08	33,12	26,90001	14,67999	39,74001	39,79999	49,82001	41,22	68,39999
	22,16	4,740002	11,12	7,98	11,96	12,64	14,6	10	5,959999	8,620003
	15,2	7,48	4,18	5,120003	4,139999	3,66	9,359997	3,360001	4,900002	3,580002
Beep	28,74	6,18	8	16,62	4,779999	5,220001	7,659996	3,100006	8,939995	3,100006
	26,5	2,260002	27	21,38	14,24	16,32	28,72	8,919998	14,08	12,88
	35,96	27,64	15,74	4,379997	4,020004	13,6	6,579994	3,340012	10,67999	4,699997
	32,5	10,88	7,640007	6,539993	6,68	8,440002	20,52	3,420006	3,759995	4,959999
	15,1	6,860001	9,039997	3,540001	6,380005	8,5	10,46	7,199997	7,860001	2,18
	47,1	30,2	25,94	31,93999	20,86	21,72	3,419998	10,02	6,539993	21,90001
	19,68	16,16	7,740005	7,579994	14,42001	6,879997	4,659996	9,380005	4,159996	6,860001
	35,02	18,92	12,22	11,26	7,959999	16,18	3,259995	6,800003	4,880005	5,539993
DL	42,84	29,94	19,18	8,139999	14,5	16,64	17,64	10,42	15,92	13,60001
	21,02	6,68	5,280003	7,239998	7,360001	5,059998	7,020004	15,93999	5,840004	10,12
	42,14	7,879997	4,440002	17,72	9,539993	9,480003	6,32	3,339996	1,479996	12,36
	35,64	24,48	5,040001	5,040001	35,58	12,86	13,88	14,14	16,23999	17,94
	19,64	7,240005	7,519997	11,96	6,340004	7,32	5,939995	3,099998	3,960007	10,01999
	49,4	4,939995	2,919998	28,56001	11,3	21,68001	20,87999	25,76001	2,459991	3,559998
	19,36	8,48	1,539997	5,940002	3,059998	6,640003	3,699997	5,480003	13,6	7,440002
EL	29,64	14,22	5,439995	9,040001	7	12,8	8,559998	6,559998	8,600006	7,719994
	45,72	6,860001	11,93999	19,3	21,22	15,17999	20,24001	9,919998	11,84	9,460007
	21,88	23,94	5,239998	13,7	5,419998	5,239998	7,260002	5,879997	7,080002	4,400002
	12,22	4,199997	7,100002	3,16	5	3,299999	3,939999	3,100002	9,200001	3,659996
	38,6	16,48	3,480003	5,519997	24,66	12,82	29,3	8,660004	13,56	11,02

REFERENCIAS A LAS FUENTES

[1] Caudell, T. P., & Mizell, D. W. (2003). *Augmented reality: an application of heads-up display technology to manual manufacturing processes*. 659–669 vol.2. https://doi.org/10.1109/HICSS.1992.183317

[2] Sutherland, I. E. (n.d.). *A head-mounted three dimensional display**.

[3] Krueger, M. W., Gionfriddo, T., & Hinrichsen, K. (1985). *VIDEOPLACE-An~2~gIi_t*.

[4] Rosenberg, L. B. (1993). Virtual fixtures: perceptual tools for telerobotic manipulation. *1993 IEEE Annual Virtual Reality International Symposium*, 76–82. https://doi.org/10.1109/VRAIS.1993.380795

[5] Amores, J., Benavides, X., & Maes, P. (n.d.). ShowMe: A Remote Collaboration System that Supports Immersive Gestural Communication. *Crossings*. https://doi.org/10.1145/2702613.2732927

[6] Cirulis, A., & Ginters, E. (2013). Augmented reality in logistics. *Procedia Computer Science*, *26*, 14–20. https://doi.org/10.1016/J.PROCS.2013.12.003

[7] *Augmented reality training for BMW assembly line workers - YouTube*. (n.d.). Retrieved September 2, 2021, from https://www.youtube.com/watch?v=V0WrCf8PbEk&ab_channel=Capgemini

[8] *Depth Cues in the Human Visual System*. (n.d.). Retrieved September 1, 2021, from http://www.hitl.washington.edu/projects/knowledge_base/virtual-worlds/EVE/III.A.1.c.DepthCues.html

[9] https://developers.google.com/ar/develop/java/depth/introduction

[10] *IEEE Xplore Full-Text PDF:* (n.d.). Retrieved September 2, 2021, from https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8115410

[11] Cipiloglu, Z., Bulbul, A., & Capin, T. (n.d.). *A Framework for Enhancing Depth Perception in Computer Graphics*. https://doi.org/10.00

[12] Heinrich, F., Bornemann, K., Lawonn, K., & Hansen, C. (n.d.). *Depth Perception in Projective Augmented Reality: An Evaluation of Advanced Visualization Techniques*. https://doi.org/10.1145/3359996.3364245

[13] *Kinect for Windows Retail Clothing Scenario Video - YouTube*. (n.d.). Retrieved September 2, 2021, from https://www.youtube.com/watch?v=Mr71jrkzWq8&ab_channel=KinectforWindows

[14] Kjaerside, K., Kortbek, K. J., Hedegaard, H., & Grønbaek, K. (2005). *Central European Multimedia and Virtual Reality Conference (2005) ARDressCode: Augmented Dressing Room with Tag-based Motion Tracking and Real-Time Clothes Simulation*.

REFERENCIAS DE LAS IMÁGENES

(1) *Espada de Damocles | IDIS*. (n.d.). Retrieved September 2, 2021, from https://proyectoidis.org/espada-de-damocles/

(2) *glowing-hockey-puck-4 | Faze*. (n.d.). Retrieved September 2, 2021, from https://faze.ca/scott-feschuk-hockey-night-in-canada/glowing-hockey-puck-4/

(3) *Creator of NFL's yellow 1st-down line inducted to Inventors HOF*. (n.d.). Retrieved September 2, 2021, from https://247sports.com/nfl/dallas-cowboys/Article/Creator-of-NFLs-yellow-1st-down-line-Stan-Honey-inducted-to-National-Inventors-Hall-of-Fame-117969941/

(4) *▷ Así lucirá Google Maps con soporte a Realidad Aumentada AR » ERdC*. (n.d.). Retrieved September 2, 2021, from https://elrincondechina.com/noticias/noticias-de-moviles-smartphones/asi-lucira-google-maps-con-soporte-a-realidad-aumentada-ar/

(5) *THE ACCOMMODATION-VERGENCE CONFLICT AND HOW IT AFFECTS YOUR KIDS (AND YOURSELF) – VR Doctors*. (n.d.). Retrieved September 2, 2021, from http://vrdoctors.net/the-accommodation-vergence-conflict-and-how-it-affects-your-kids-and-yourself/

(6) *Depth Cues in the Human Visual System*. (n.d.). Retrieved September 1, 2021, from http://www.hitl.washington.edu/projects/knowledge_base/virtual-worlds/EVE/III.A.1.c.DepthCues.html

(7) *Perspectiva aérea en el arte – HiSoUR Arte Cultura Historia*. (n.d.). Retrieved September 2, 2021, from https://www.hisour.com/es/aerial-perspective-in-art-51697/

(8) *Psicología de la Percepción - UCM: Percepción de la profundidad y el tamaño de los objetos*. (n.d.). Retrieved September 2, 2021, from http://psicologiapercepcion.blogspot.com/p/percepcion-de-la-profundidad-y-el.html

(9) Heinrich, F., Bornemann, K., Lawonn, K., & Hansen, C. (n.d.). *Depth Perception in Projective Augmented Reality: An Evaluation of Advanced Visualization Techniques*. https://doi.org/10.1145/3359996.3364245

(10)