UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Desarrollo de una aplicación para entrenamiento motor basada en sistemas mocap

Trabajo Fin de Grado

Grado en Tecnología Digital y Multimedia

AUTOR/A: Fontes Albeza, Alejandro

Tutor/a: Rey Solaz, Beatriz

Cotutor/a: Monzó Ferrer, José María

CURSO ACADÉMICO: 2022/2023

Resumen

Este trabajo se enfoca en el desarrollo de una aplicación interactiva que utiliza tecnología de captura de movimiento en tiempo real de OptiTrack y el motor de videojuegos Unity. La aplicación permite a los usuarios participar en ejercicios físicos y ver cómo sus movimientos se reflejan en personajes virtuales. La interfaz consta de cuatro partes clave: pantalla de carga, selección de personaje, elección de ejercicio y una pantalla final.

Para lograr esto, se llevaron a cabo varios pasos, incluyendo el diseño e integración de escenas, la incorporación de efectos de sonido y la programación de scripts para controlar la interacción. Se puso un fuerte énfasis en la recepción y procesamiento de datos capturados, donde la persona usa un traje especial y el *software* Motive captura sus movimientos, transmitiendo estos datos en tiempo real a Unity a través de un *plugin*.

Se realizaron pruebas exhaustivas para garantizar la precisión de la captura de movimiento y la usabilidad de la interfaz. Se evaluó visualmente el correcto funcionamiento de la captura de movimiento y la interfaz. Además, se resolvieron desafíos técnicos para asegurar una comunicación efectiva entre los dos sistemas.

Este proyecto contribuye a los Objetivos de Desarrollo Sostenible (ODS) 3 y 4, promoviendo la salud y el bienestar a través de la participación en ejercicios físicos, y proporcionando una educación de calidad a través de una plataforma interactiva que enseña y mejora la técnica de movimiento. En resumen, esta aplicación innovadora se alinea con el avance tecnológico y el bienestar humano.

Resum

Aquest treball es centra en el desenvolupament d'una aplicació interactiva que utilitza la tecnologia de captura de moviment en temps real d'OptiTrack i el motor de videojocs Unity. L'aplicació permet als usuaris participar en exercicis físics i veure com els seus moviments es reflecteixen en personatges virtuals. La interfície consta de quatre parts clau: pantalla de càrrega, selecció de personatge, elecció d'exercici i una pantalla final.

Per a aconseguir-ho, es van dur a terme diversos passos, incloent el disseny i la integració d'escenes, la incorporació d'efectes de so i la programació de scripts per controlar la interacció. Es va posar un fort èmfasi en la recepció i el processament de dades capturades, on la persona utilitza un vestit especial i el sistema Motive captura els seus moviments, transmetent aquestes dades en temps real a Unity a través d'un *plugin*.

Es van realitzar proves exhaustives per garantir la precisió de la captura de moviment i la usabilitat de la interfície. Es va avaluar visualment el correcte funcionament de la captura de moviment i la interfície. A més, es van resoldre desafiaments tècnics per assegurar una comunicació efectiva entre els dos sistemes.

Aquest projecte contribueix als Objectius de Desenvolupament Sostenible (ODS) 3 i 4, promocionant la salut i el benestar a través de la participació en exercicis físics i proporcionant una educació de qualitat a través d'una plataforma interactiva que ensenya i millora la tècnica de moviment. En resum, aquesta aplicació innovadora s'alinea amb l'avanç tecnològic i el benestar humà.

Abstract

This work focuses on the development of an interactive application that utilizes real-time motion capture technology from OptiTrack and the Unity game engine. The application allows users to engage in physical exercises and see how their movements are reflected in virtual characters. The interface consists of four key parts: loading screen, character selection, exercise selection, and a final screen.

To achieve this, several steps were taken, including scene design and integration, the incorporation of sound effects, and scripting to control interaction. Strong emphasis was placed on receiving and processing captured data, where the person wears a special suit, and the Motive system captures their movements, transmitting this data in real-time to Unity through a *plugin*.

Comprehensive testing was conducted to ensure the accuracy of motion capture and interface usability. The proper functioning of motion capture and the interface was visually evaluated. Additionally, technical challenges were resolved to ensure effective communication between the two systems.

This project contributes to Sustainable Development Goals (SDGs) 3 and 4 by promoting health and well-being through physical exercise participation and providing quality education through an interactive platform that teaches and improves movement technique. In summary, this innovative application aligns with technological advancement and human well-being.

Índice

Índice de Figuras

Capítulo 1. INTRODUCCIÓN

La captura de movimiento es uno de los sistemas tecnológicos más modernos a la hora de crear contenido relacionado con el mundo del entretenimiento en áreas como la animación, los videojuegos y aplicaciones o incluso la mezcla de estas con el mundo cinematográfico. La captura de movimiento permite grabar y reproducir movimiento de seres humanos prácticamente a la perfección y de esta forma poder trasladar dichos movimientos a personajes ficticios creados digitalmente por ordenador.

En este trabajo, se diseña y desarrolla una aplicación para entrenamiento motor en Unity combinada con la captura de movimiento de OptiTrack en tiempo real. Gracias a la combinación de ambos *softwares* se crea una experiencia interactiva e inmersiva para el usuario final.

1.1 Contexto y Justificación

Hoy en día, los videojuegos y las aplicaciones que se encuentran en el mercado se han popularizado en la sociedad y se han convertido en una gran herramienta del humano a la hora de crear e innovar. Pero hay que destacar que en la mayoría de los sistemas y entornos virtuales se opta por utilizar entradas como son el ratón, el teclado o controladores de juego.

En cambio, con la captura de movimiento se consigue que el usuario se sienta dentro del entorno, no importa que sea un videojuego o en este caso una aplicación enfocada al entrenamiento motor. De esta forma, se logra clonar el movimiento de una persona en tiempo real y trasladarlo a un personaje animado o ficticio. Además, el movimiento capturado se puede guardar en un fichero y utilizarlo posteriormente para realizar comprobaciones métricas relacionadas con entrenamiento motor en el caso de este trabajo.

1.2 Objetivos del trabajo

El objetivo principal del trabajo es realizar una aplicación de entrenamiento motor en Unity y que sea capaz de leer el movimiento del usuario en tiempo real por medio del sistema OptiTrack de forma precisa y segura, y mostrar al usuario dicho movimiento en un personaje virtual 3D.

Para llevar a cabo todo esto es necesario seguir una serie de pasos:

1. Entender y comprender el funcionamiento del sistema OptiTrack con el *software* Motive y así conseguir crear un esqueleto base para realizar la captura de movimiento.
2. Configurar y calibrar el sistema de captura de movimiento en OptiTrack para poder garantizar el desarrollo de la aplicación de forma precisa y consistente.
3. Desarrollar la comunicación entre los dos sistemas Unity y Motive que se utilizan en el proyecto, de forma segura y exacta.
4. Añadir al proyecto la lectura en tiempo real del movimiento en Unity, consiguiendo una sincronización perfecta.
5. Controlar el movimiento de un avatar en Unity en base a los datos de movimiento recibidos.
6. Importar y aplicar los movimientos capturados, exportados en formato *fbx*, al personaje instructor integrado en la aplicación para facilitar la ejecución del ejercicio por parte del usuario.
7. Diseñar y desarrollar un interfaz de usuario para la aplicación donde se permita seleccionar entre distintos ejercicios.
8. Realizar pruebas y así poder validar la aplicación y comprobar su funcionalidad y rendimiento.

1.3 Fases del Proyecto

Con los objetivos principales del proyecto planteados en el punto anterior, a continuación, se listan y se describen las fases por las que se va a llevar a cabo el proyecto:

1. **Funcionamiento del sistema OptiTrack:** Para entender mejor cómo funciona la tecnología relacionada con la captura de movimiento, concretamente con el sistema de OptiTrack, y el desarrollo de aplicaciones en Unity, se investigará de forma exhaustiva por Internet y revisando proyectos realizados en la universidad durante el grado.
2. **Configuración de la comunicación entre sistemas:** Se configurará el sistema de captura OptiTrack para que se pueda comunicar con el motor de Unity en tiempo real.
3. **Desarrollo del entorno virtual en Unity:** El usuario será capaz de escoger un personaje, un movimiento y por último moverse por una sala donde dispondrá de cierto tiempo para imitar el ejercicio del personaje instructor.
4. **Pruebas y refinamiento del sistema:** Mediante pruebas en la sala y desde casa se conseguirá desarrollar la aplicación de forma segura y precisa.
5. **Análisis de resultado y futuras mejoras:** Se analizarán los resultados de dichas pruebas y a partir de ahí se extraerán conclusiones respecto a los objetivos planteados desde un principio. Además, se valorarán posibles mejoras a futuro.

Definidas las metodologías que se van a utilizar a la hora de desarrollar el proyecto, con estas se espera conseguir una aplicación segura y fiable, además de conseguir resultados óptimos para el mundo de la captura de movimiento y la realidad virtual.

1.4 Objetivos de Desarrollo Sostenible

El trabajo contribuye al Objetivo de Desarrollo Sostenible 3: **Salud y Bienestar**, al proporcionar una aplicación que fomenta la actividad física y el ejercicio. Al permitir que los usuarios realicen movimientos guiados por un personaje virtual, la aplicación puede ser una herramienta efectiva para mejorar la salud y el bienestar general. Los usuarios pueden acceder a ejercicios específicos, recibiendo orientación en tiempo real y retroalimentación visual. Esto respalda directamente la promoción de estilos de vida saludables y la prevención de enfermedades asociadas con la inactividad.

Asimismo, el trabajo se alinea con el Objetivo de Desarrollo Sostenible 4: **Educación de Calidad**, al ofrecer a los usuarios una plataforma de aprendizaje interactiva. A través de la aplicación, los usuarios pueden aprender sobre diferentes ejercicios y técnicas de movimiento de manera práctica. La combinación de la orientación visual y la posibilidad de realizar seguimiento de los movimientos permite a los usuarios mejorar sus habilidades y conocimientos en cuanto a la realización de ejercicios físicos. Esto demuestra cómo la tecnología puede contribuir a la educación y al empoderamiento en el ámbito de la salud y el bienestar.

Capítulo 2. FUNDAMENTOS TEÓRICOS

Antes de comenzar a explicar con detalles cómo se ha desarrollado la aplicación es necesario entender y comprender varios conceptos. En primer lugar, en este apartado se definirá la tecnología de la captura de movimiento y sus principios básicos. Además, se profundizará en el apartado de OptiTrack, donde se explicará con detalle el funcionamiento de este tipo de sistema. Por último, se dedicará un apartado exclusivo al motor de videojuegos Unity. Se realizará una pequeña introducción de este motor y además se explorarán las diferentes herramientas que se utilizan en él, para que de esta forma se entienda mejor la base del proyecto.

2.1 Captura de movimiento

La captura de movimiento es la técnica, que se está popularizando hoy en día, que recoge los datos exactos del movimiento de una persona o un objeto. Esta tecnología es capaz de caracterizar y dar vida a personajes virtuales en 3D que se han creado por ordenador, incluso se puede llegar al punto de imitar acciones y gestos que solo una persona puede realizar; es decir, crear un personaje virtual único. Por otro lado, en dichos registros solo se guardan los movimientos realizados, en ningún momento se guarda la apariencia del actor que lo realiza. Finalmente, al registrar los datos estos se pueden trasladar y asignar a un modelo 3D en otro entorno virtual, por ejemplo, a la hora de realizar videojuegos, aplicaciones o películas de animación [1]. A continuación, en la figura 1 se puede observar un ejemplo clásico de la película **El Señor de los anillos** donde se está utilizando la captura de movimiento para realizar movimientos realistas a los personajes de la película.

Figura 1. Ejemplo de sistema Mocap [2]

2.1.1 Principios básicos de la captura de movimiento

En primer lugar, hay que entender que hay diferentes formas de capturar el movimiento y que según las aplicaciones que se usen y las técnicas que se usen se tendrá una base u otra. Para el caso de este trabajo, hay que entender que para capturar el movimiento es necesario disponer de una sala con cámaras especiales, concretamente las cámaras de OptiTrack **Flex 3**, además de un traje donde se adhieren diferentes marcadores según el esqueleto y el movimiento que se desee realizar.

Para poder grabar el movimiento correctamente, es necesario calibrar la posición y la orientación de las cámaras respecto al suelo donde se van a realizar las mediciones. Este proceso se realiza gracias a unos dispositivos que disponen de diferentes marcadores colocados exactamente en una posición, los cuales funcionan como sensores que se detectan mediante las cámaras.

Por último, se debe tener claro qué tipo de movimiento se desea realizar y a qué personaje o a qué objeto va a ir implementado. Teniendo claro esto, ya se podría realizar la captura en tiempo real y se habrá explicado los principios de la captura de movimiento [3].

En pocas palabras, la captura de movimiento es una técnica que consiste en detectar y rastrear señales colocadas en el cuerpo de una persona o en un objeto. Estos movimientos son reconocidos por cámaras o sensores infrarrojos, controlando así cada detalle con precisión. La calibración y la corrección de errores son esenciales para garantizar la precisión. Se procesan y se filtran los datos registrados para obtener los mejores y más detallados resultados. La tasa de muestreo determina cuántos datos se recopilan por segundo y afecta a la precisión y a la cantidad de información. Una vez que se reciben los datos, se combinan con un motor de animación o juego para animar un personaje o aplicar cambios a un modelo 3D. Estos datos pueden transmitirse en tiempo real o mediante ficheros en formatos *fbx* que se registran y se quedan guardados. La captura de movimiento se usa ampliamente en campos como la animación, el entretenimiento y la investigación científica, y puede simular el movimiento de forma real y precisa.

2.1.2 Tecnología OptiTrack

La tecnología OptiTrack es un sistema comercial de captura de movimiento implementada en diversas áreas tecnológicas hoy en día como son el mundo de los videojuegos, la animación, el entretenimiento, la realidad virtual o la investigación científica. Está tecnología fue desarrollado por la empresa **Natural Point**, donde los creadores destacan que OptiTrack se define por ser un sistema de alta precisión, calidad y seguridad para capturar y registrar movimientos tanto de personas como de objetos [4].

Este sistema, como ya se ha comentado en el punto anterior, utiliza diferentes cámaras infrarrojas y sensores reflectantes para así lograr la captura de movimiento de dichos sensores. Una vez las cámaras detectan estos datos, los trasladan al *software* donde se está reflejando exactamente el movimiento en tiempo real. Una de las mayores ventajas que dispone OptiTrack es la captura de movimiento múltiple, es decir, que en la misma escena

a la vez se pueden estar capturando en tiempo real movimientos de diferentes personas como de objetos. Esto hace que OptiTrack se convierte en uno de los sistemas más flexibles y versátiles a la hora de crear escenas para después trasladarlo a una película, videojuego, aplicación, etc.

Por otro lado, esta tecnología destaca gracias al *software* proporcionado y a su flexibilidad a la hora de realizar procesamiento de los datos capturados, además, esta permite comunicarse fácilmente y trasladar datos con precisión a diferentes motores de juegos, como es el caso de Unity.

En la siguiente figura 2, se puede observar el funcionamiento del *software* de captura de movimiento Motive.

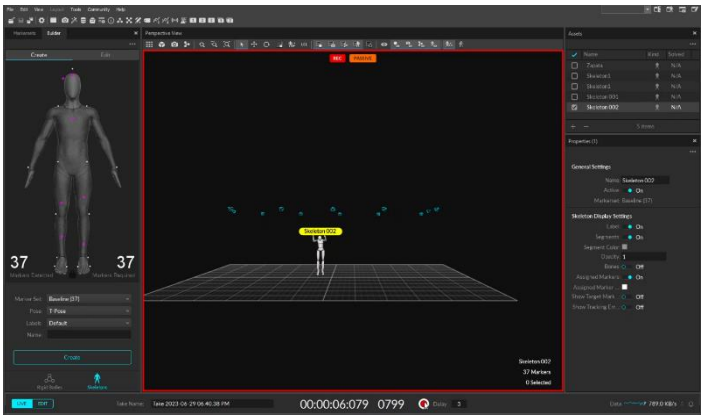

Figura 2. Captura de movimiento en Motive

2.2 Unity

A continuación, en este apartado se desarrollará una pequeña introducción al motor de videojuegos conocido en su industria Unity y que tipo de herramientas de desarrollo se utiliza en esta tecnología.

2.2.1 Introducción a Unity

Unity es un motor de videojuegos multiplataforma 2D y 3D desarrollado por **Unity Technologies** el 30 de mayo de 2005. Se ha desarrollado y programado en *C++* y *C#* para los sistemas operativos **Windows**, **Mac OS** y **Linux**.

El lenguaje utilizado a la hora de crear aplicaciones y videojuegos es *C#* o *C Sharp* en su versión inglesa o *C#*. Es un lenguaje de programación diseñado por **Microsoft** que principalmente está orientado a objetos. Esto significa que se enfoca en aquellos objetos que el programador desee manipular en vez de centrarse en la lógica que se debe de utilizar para dicha manipulación [5].

En Unity se incluye dentro de la propia aplicación, un editor de código, un editor visual, un motor de física, herramientas para la renderización y muchos más componentes para poder componer mundos virtuales.

Por otro lado, este motor incluye otras funcionalidades avanzadas como son la iluminación de escenas, efectos especiales *VFX*, audio y un gran soporte de redes para realizar plataformas multijugador. Además, a la hora de estar creando escenas y espacios virtuales los usuarios pueden realizar cambios y variaciones en sus proyectos y estas se verán reflejadas al instante. Esto facilita y acelera el proceso de desarrollo e iteración. A continuación, se muestra en la figura 3 el espacio de trabajo que utiliza Unity para crear proyectos [6].

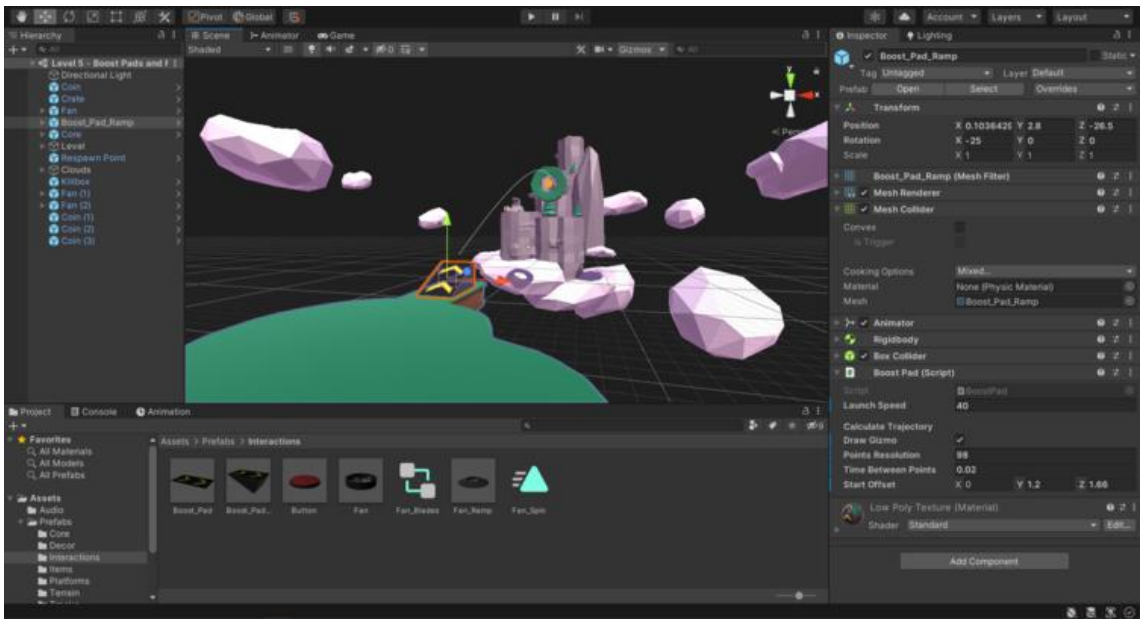

Figura 3. Espacio de trabajo en Unity [7]

2.2.2 Herramientas de desarrollo en Unity

En cuanto a las herramientas de desarrollo en Unity, se encuentran varias importantes que se explican a continuación.

En primer lugar, el editor de Unity, donde el usuario puede crear diferentes escenas, materiales, objetos, luces, animaciones o *scripts* para poder aplicárselo a componentes en la escena y que estos interactúen al gusto del usuario. El editor de código que utiliza este motor de videojuegos es **Visual Studio**. Este editor permite una gran flexibilidad y personalización de los proyectos, facilitando así el proceso de desarrollo.

Después se encuentra el apartado de animación de Unity. Aquí se pueden crear animaciones de personajes, objetos y efectos visuales. Aquí los desarrolladores pueden crear todo tipo de *keyframes* para así lograr animaciones en los componentes de sus proyectos. En cambio, la mayoría de los desarrolladores optan por grabar o crear estos

efectos y animaciones desde otros sistemas y aplicaciones como son el caso de **Blender** o OptiTrack, ya que se tratan de tecnologías dedicadas a este tipo de funcionalidades.

También, Unity incluye su propio motor de física que simula comportamientos realistas entre objetos y el mundo virtual. Desde aquí se pueden aplicar fuerza, gravedad, rozamiento e incluso colisiones. Esto una de las mayores ventajas que dispone Unity a la hora de realizar proyectos realistas y originales.

Por último, dispone de una tienda online llamada **Asset Store**, donde se pueden descargar ítems de pago o gratis para el proyecto tales como materiales concretos, objetos específicos automóviles, árboles, animales, etc. Además, al tratarse de una tienda online vinculada con Unity es muy sencillo importar dichos objetos a cualquier proyecto.

En la siguiente figura 4 se puede observar la pantalla principal de dicha tienda online para hacer uso de ella en los proyectos que se requiera.

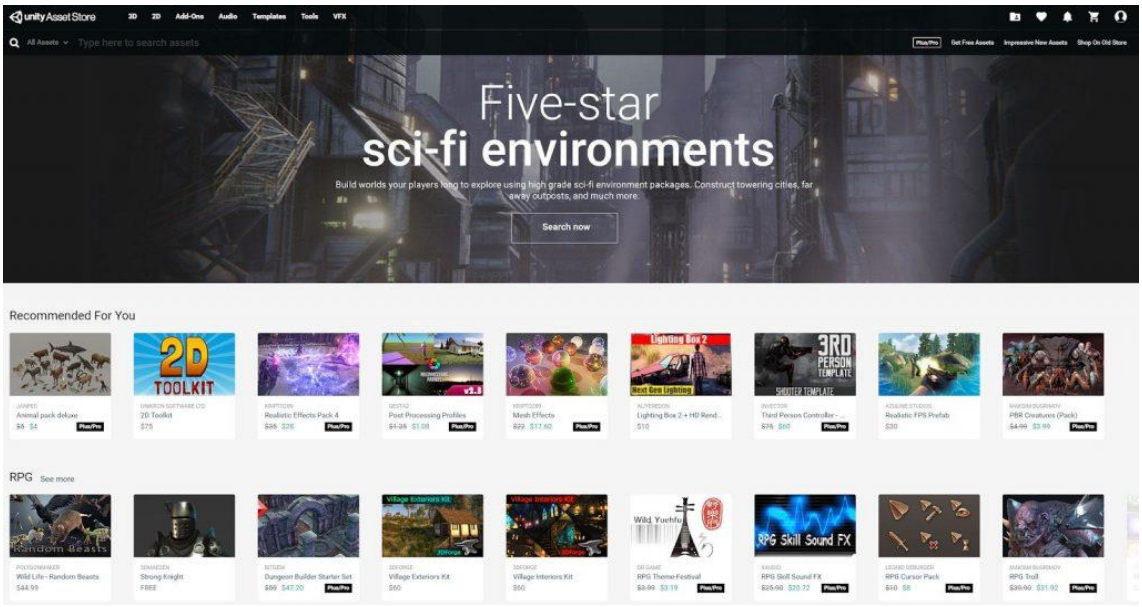

Figura 4. Unity Asset Store

Capítulo 3. DISEÑO DE LA APLICACIÓN

En los siguientes apartados se van a abordar los diferentes pasos de diseño y configuraciones que se han llevado a cabo a la hora de realizar la aplicación además de la captura de movimiento en OptiTrack y su posterior integración en el motor de videojuegos de Unity.

3.1 Arquitectura del Sistema

A la hora de diseñar la arquitectura del sistema se tienen varias cosas en consideración como es la captura de movimiento con OptiTrack, la integración con Unity y su interacción en tiempo real. Esta arquitectura se basa en cómo se estructura el proyecto y cómo se va a diseñar. Básicamente es una descripción que se establece para poder saber cómo se organizan y conectan los diferentes elementos del proyecto.

3.1.1 Materiales

En primer lugar, la **ETSIT** dispone de una sala donde están instaladas y configuradas las cámaras **Flex 3** que vienen con el sistema de OptiTrack. Todas deben apuntar a un mismo punto, que es donde se va a situar la supuesta persona que va a utilizar la aplicación. También, hay que tener en cuenta que el *software* que utiliza OptiTrack con el nombre Motive, su suscripción es de pago. Pero gracias a la **Universitat Politècnica de València** se ha podido llevar a cabo el proyecto sin problemas.

En segundo lugar, se encuentra un traje, el cual dispone de vestimenta para todas las partes del cuerpo dependiendo de las animaciones que se deseen generar; además de unos marcadores llamados **Marcadores Rígidos** o **Activos**. Estos marcadores son elementos pequeños y ligeros ya que van sujetos al traje y deben de ser fáciles de transportar para así poder realizar movimientos de todo tipo. Funcionan reflejando la luz infrarroja emitidas por las cámaras, comentadas anteriormente. Las cámaras reciben la luz reflejada, lo cual permite detectar la posición y la orientación de los marcadores situados en el traje. Es necesario destacar que según el tipo de esqueleto que se quiera animar se utilizarán más o menos marcadores. Para el caso de este proyecto se ha optado por un esqueleto estándar ya que es el más sencillo de configurar y para después trasladar la información a Unity, se llama **Baseline 37**; y como bien dice su nombre, contiene exactamente 37 marcadores distribuidos por todo el cuerpo, desde la cabeza hasta los pies. En la figura 5 se puede ver el espacio de trabajo del *software* Motive donde se puede visualizar el esqueleto que se ha usado en el proyecto con sus respectivos marcadores fundamentales.

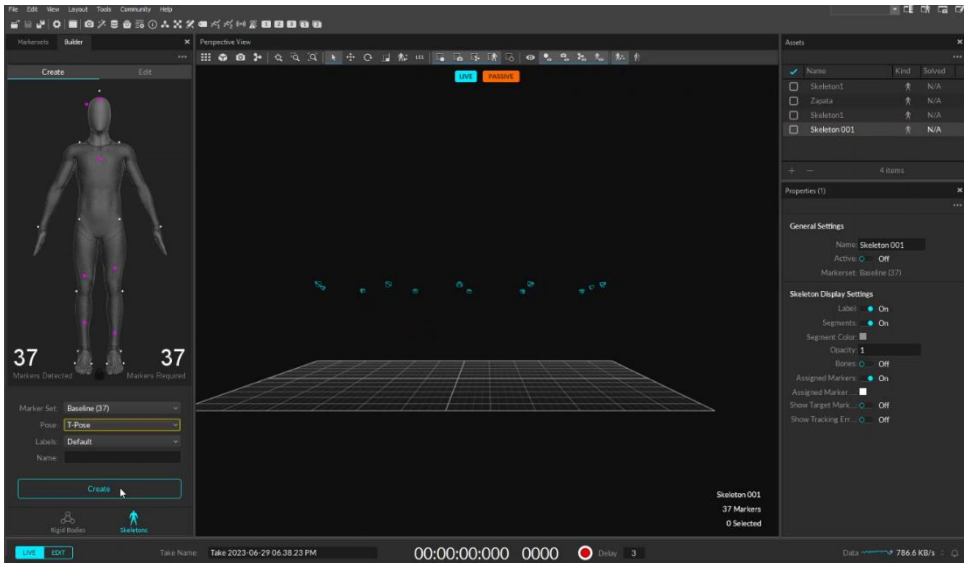

Figura 5. Espacio de trabajo en Motive

Pero antes de estos pasos es necesario calibrar la zona donde se van a grabar los movimientos del personaje y donde se va a situar el usuario que va a hacer uso de la aplicación. Para ello se encuentran diferentes ventanas en el *software* de Motive las cuales son la de calibración, la de creación de esqueleto, la de captura de movimiento y por último la ventana de edición donde se podrá modificar algún movimiento en caso de ser necesario. Además, será en esta ventana donde se exportará los movimientos del personaje instructor de la aplicación.

Para dicha calibración se hará uso de dos objetos, el calibrador de cámaras **CW-500 Calibration Wand Kit** y el calibrador de suelo **CS-100 Calibration Square** [8]. Estos dos objetos son fundamentales ya que son los encargados de dar información al *software* del volumen de la sala donde se va a capturar los movimientos y para saber la posición del suelo, calculando la inclinación, la orientación y la altura respecto a las cámaras. En las siguientes imágenes, figura 6 y figura 7, se muestran dichos calibradores de cámaras.

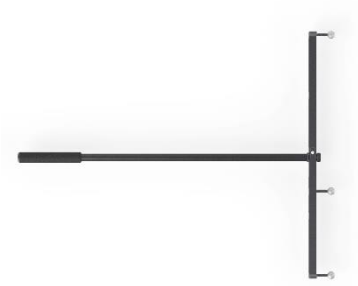

Figura 6. CW-500 Calibration Wand

Figura 7. CS-100 Calibration Square

Por último, para llevar a cabo el proyecto se hace uso del motor de videojuegos conocido Unity el cual es gratuito y se puede descargar cualquier usuario desde su página web. No hace falta explicar con detalle este motor ya que en los primeros puntos ya se ha detallado toda la información fundamental. Para poder comunicar el *software* de Motive con el motor de videojuegos de Unity se ha utilizado un *plugin* que crearon ambas empresas para poder llevar a cabo dicha comunicación. Este *plugin* se instala dentro del proyecto de Unity y con él es sencillo comunicar ambas aplicaciones ya que funciona mediante *IP*.

En este caso, para el proyecto se ha seguido adelante con un solo ordenador donde en este se ejecutaban ambas aplicaciones para realizar la comunicación y poder llegar a los objetivos comentados en el capítulo 1 del proyecto.

Finalmente, se descargaron personajes y animaciones de la página web de **Mixamo** [9] y se pudo empezar a realizar pruebas con el proyecto y así conseguir avanzar óptimamente durante el transcurso del desarrollo de la aplicación. Gracias a dicha aplicación, se pueden descargar tanto los personajes que han sido implementados definitivamente en la aplicación; el chico, la chica y el profesor, como animaciones de prueba para el personaje instructor antes de grabar las definitivas en la sala de captura de movimiento de la **Universitat Politècnica de València.**

3.1.2 Diagrama de bloques

En la Figura 8 se presenta un esquema que organiza los distintos componentes del trabajo en sus respectivas partes. En este esquema se muestra la disposición del traje equipado con los sensores de captura de movimiento. Se puede observar también la configuración del ordenador que ha sido utilizado para ejecutar simultáneamente tanto los programas de captura de movimiento de OptiTrack Motive como la aplicación desarrollada en el motor de videojuegos Unity. La figura ilustra cómo la información capturada se transfiere y se integra en la aplicación creada en Unity, la cual se ejecuta de manera sincronizada con el programa de captura de OptiTrack Motive. Este esquema visual ofrece una comprensión clara de cómo se relacionan los diferentes elementos y tecnologías en el proceso de captura y reproducción de movimiento en tiempo real.

Figura 8. Diagrama de bloques

3.2 Captura de movimiento con OptiTrack

En el siguiente apartado se explicarán varios aspectos relacionados con la captura de movimiento con OptiTrack, como son **la configuración del sistema de captura** y **la calibración y sincronización.** También se profundizará en el apartado de **creación de esqueleto** y **configuración y colocación de los marcadores.** En ellas se abordará como se debe configurar y calibrar el *software* y la sala, además de cómo deben estar colocados los marcadores y de qué tipo de esqueleto se va a hacer uso para desarrollar el proyecto.

3.2.1 Configuración y Calibración del Sistema de Captura

Para poder configurar correctamente el sistema de captura, es necesario instalar todas las cámaras en puntos específicos y con un ángulo concreto para que estas apunten a la zona donde se va a colocar la persona con el traje. Para ello un método óptimo, sería colocar el calibrador de suelo en un punto y comprobar que, en el apartado de cámaras, todas las cámaras detectan dicho punto.

Una vez comprobado este paso, lo siguiente será realizar el calibrado de las cámaras. Para ello, en la aplicación de Motive hay que situarse en el apartado de ventanas y se selecciona la opción de captura. Una vez aquí se pulsa el botón de calibrar y en la parte de debajo de la pantalla se visualiza todo lo que están detectando las cámaras en tiempo real. Además, en la parte de abajo izquierda del programa se muestran los datos que se están capturando en cada cámara. Para una óptima captura es necesario que todas las cámaras lleguen a la cantidad de 3000 puntos capturados, esto se puede ir viendo a medida que se está realizando el calibrado. Ahora se hace uso del calibrador de cámaras **CW-500 Calibration Wand Kit** y se girará y se harán movimientos intentando dibujar en el aire el número 8 por todas las zonas de la sala. Esto hará que las cámaras detecten todos los puntos del calibrador y sean capaces de detectar la información fundamental de la zona donde se va a realizar la captura correctamente [10]. En las figuras 9 y 10 se puede observar, cómo se debe realizar el calibrado con el instrumento nombrado anteriormente y que se visualiza en la pantalla de Motive respectivamente.

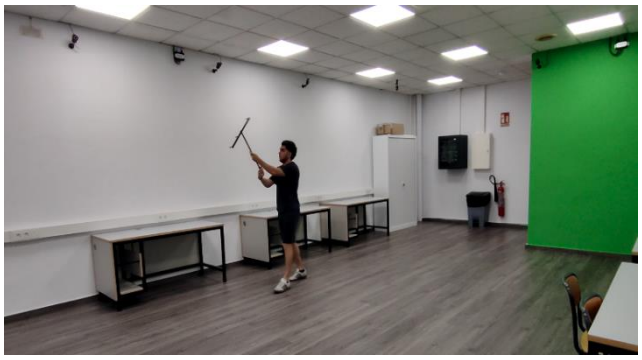

Figura 9. Calibrado de cámaras

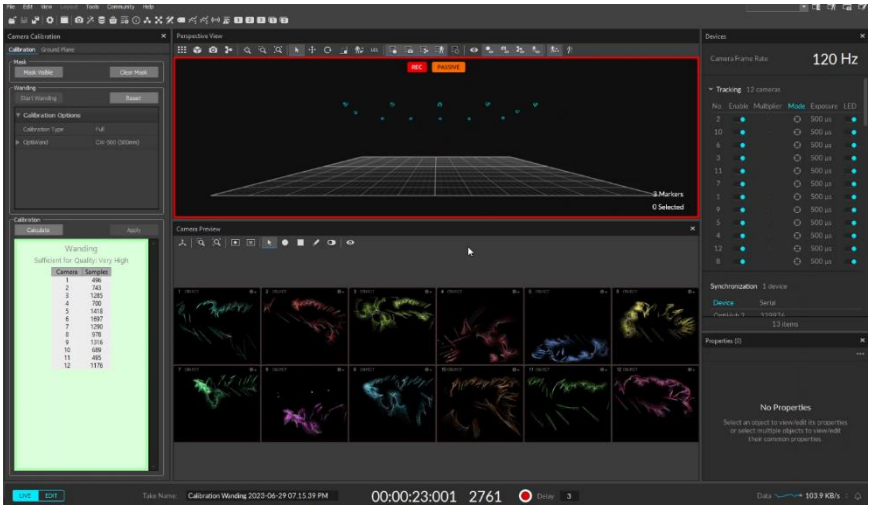

Figura 10. Calibrado de cámaras en Motive

Una vez se haya llegado a los 3000 puntos en cada cámara se pulsa en finalizar y se mostrará un mensaje diciendo que la captura ha sido excepcional. En caso contrario, será necesario realizar los pasos anteriores y volver a realizarlo. Si todo se ha hecho correctamente, se muestra en la ventana de espacio de trabajo que la aplicación Motive ha detectado todas las cámaras instaladas, pero sigue faltando una cosa ya que, al realizar este paso, dentro del espacio de trabajo de Motive las cámaras tienen una orientación errónea respecto al suelo; y esto se debe a que aún no se ha realizado la calibración del punto central situado en el suelo.

El siguiente paso es calibrar el suelo del proyecto. Al pulsar el botón de finalizar la calibración de las cámaras, automáticamente la aplicación cambiará de disposición y se puede ver como aparece un botón de calibración de suelo. A continuación, se pone el calibrador correspondiente de suelo CS-100 Calibration Square en el punto exacto donde se vaya a situar la persona ya que es el punto más centrado de la sala. Una vez hecho, se pulsa en el botón de calibración en la aplicación de Motive y tras unos segundos el espacio de trabajo se calibrará correctamente. Y, por tanto, se visualiza en pantalla exactamente el espacio de la sala de grabación.

Por último, es importante destacar que si la sala dispone de aire acondicionado este debe permanecer apagado ya que puede perjudicar a la calibración y por tanto estropear el desarrollo del trabajo.

3.2.2 Colocación y configuración de los marcadores. Esqueleto Motive

Una vez calibrado el área donde se va a llevar a cabo el proyecto, el siguiente paso es configurar un esqueleto para recibir los movimientos del actor. Para ello, hay que situarse en la ventana de *Create* y una vez aquí tener claro qué tipo de personaje se va a utilizar y qué movimientos se desean realizar. En el caso de este proyecto, se va a requerir de personajes humanos sin importar la altura, el peso o el sexo ya que se tratan de ejercicios físicos y todo ser humano hace uso de los mismo huesos y músculos sea el ejercicio que

sea y sin importar su condición física. En el *software* Motive encontramos esqueletos de todo tipo, desde los más complejos incluyendo marcadores para los dedos hasta esqueletos más sencillos incluyendo solo la parte inferior o superior del cuerpo. Teniendo esto claro, en el trabajo se va a optar por utilizar el esqueleto **Baseline (37)**, un esqueleto completo el cual requiere 37 marcadores, pero sin la necesidad de utilizar para los dedos u otras partes más complejas del cuerpo de capturar. En pantalla se muestra claramente las zonas exactas donde deben ir colocados los marcadores, destacando con color morado los puntos más importantes del esqueleto. Dichos puntos tienen que estar exactamente donde se muestra, si no, la grabación de los movimientos será errónea. En la figura 11 se puede observar cómo Motive indica claramente donde se deben de colocar los marcadores en el traje para realizar una captura de movimiento óptima con el esqueleto seleccionado.

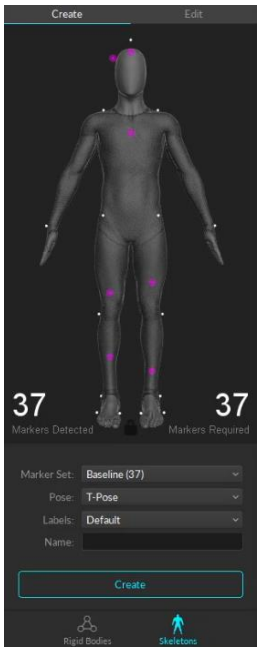

Figura 11. Marcadores esqueleto Baseline (37)

Para conseguir colocar los marcadores donde corresponden se puede optar por la ayuda de otra persona ya con el traje puesto y si no, se recomienda pegar antes los marcadores en el traje y luego ponérselo; ya que será mucho más sencillo para aquellas zonas como la espalda donde no se puede llegar con facilidad.

En el momento en el que el traje esté colocado perfectamente, la persona se coloca en el centro de la sala donde todas las cámaras puedan detectar los marcadores. De hecho, el programa mostrará en tiempo real todos los marcadores que está detectando. En el momento en el que se llegue al número correcto, en este caso al 37, el botón de crear se iluminará y se podrá crear el esqueleto. Es recomendable que antes de pulsar el botón de crear, la persona se sitúe en modo *T-pose*, en otras palabras, dibujar la letra T con las extremidades. Esto es también recomendable, ya que para el programa es más sencillo de crear un esqueleto, debido que al realizar dicha pose todos los marcadores se separan entre ellos y las cámaras los pueden detectar mejor. A continuación, en la figura 12 se muestra un ejemplo de cómo se debe situar el actor a la hora de realizar los movimientos a capturar.

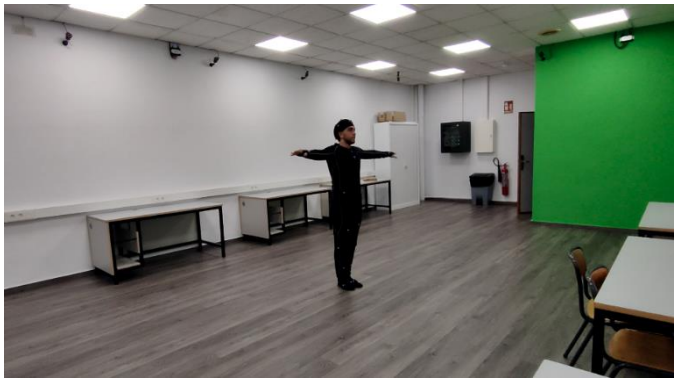

Figura 12. T-pose

Una vez comprobado que todo esté correcto, ya se puede pulsar el botón de crear el esqueleto y al instante se podrá ver en la ventana de espacio de trabajo el esqueleto seleccionado siguiendo el movimiento que esté realizando la persona en ese momento.

A partir de aquí ya se puede grabar cualquier movimiento y posteriormente hacer uso de él. En este proyecto se conseguirá pasar movimientos grabados al personaje instructor y además el movimiento del usuario que vaya a utilizar la aplicación a los personajes predeterminados del programa en tiempo real. Más adelante se abordará este tema con más detalle.

3.3 Integración con Unity

En los siguientes apartados se va a abordar detalladamente cómo se debe configurar Unity y Motive para poder realizar el traspaso de información de un *software* a otro. Para ello, se va a utilizar un *plugin* creado por OptiTrack y Unity específicamente para este tipo de trabajo.

3.3.1 OptiTrack-Unity *Plugin*

El *plugin* de OptiTrack para Unity es una extensión de *software* que permite la integración y comunicación de los datos de captura de movimiento obtenidos mediante el sistema de OptiTrack con el *software* de Motive y trasladados posteriormente al motor de videojuegos de Unity. Gracias a este *plugin* se puede conseguir la importación y el uso de los datos de movimiento capturados en tiempo real y llevarlos fácilmente a proyectos realizados en Unity; consiguiendo experiencias reales e inmersivas para los usuarios. El contenido del paquete se importa fácilmente en Unity mediante el formato *unitypackage*, el cual se trata de un fichero comprimido que contiene los ficheros necesarios. Una vez importados; los scripts se usan para definir el cliente y así poder recibir los datos del movimiento.

Las versiones que se recomienda para hacer uso de dicho *plugin* son las siguientes:

- Versión de Unity: 2017.2 / 2017.1 o superior. (2020.3+ recomendado)
- Visual Studio 2019 o la última versión redistribuible de **Visual C++**

Dentro del paquete se encuentran los *scripts* configurados para la recepción de información del *software* de Motive. También incluye complementos que hacen referencia a la biblioteca que usa dicho *plugin* para funcionar correctamente. Además, también dispone de *prefabs*, que son componentes especiales de Unity que permiten el uso de objetos complejos plenamente configurados, y escenas para poder realizar pruebas en caso de no disponer en el proyecto. En el caso de este trabajo no se ha hecho uso de estos ya que con los componentes que se disponía se podía llevar a cabo la transmisión de información de forma correcta.
Para hacer uso de este *plugin*, en la página oficial de **OptiTrack** [11] se puede encontrar guía rápida que explica todo detallado paso a paso además de información extra que para este trabajo no es necesaria.

3.3.2 Configuración de Unity y Motive para mandar y recibir datos

Para poder realizar la comunicación correctamente es necesario configurar los sistemas de una determinada forma. Primero se configura Motive ya que es el que funciona como servidor; es decir, el que transmite la información y posteriormente Unity ya que es el que recibe la información deseada.

En Motive, una vez se haya creado el esqueleto deseado y se vea en la ventana de trabajo, se selecciona la ventana de **Ajustes de Transmisión** o ***Data Streaming*** y aparecen las opciones que se pueden ver en la figura 13. Se deben dejar de la siguiente forma excepto la opción de ***Transmission Type*** que debe estar en ***Unicast***, ya que es la configuración deseada para este caso de transmisión en tiempo real. Hay que tener en cuenta cómo se llama el esqueleto creado ya que posteriormente en Unity se debe de introducir una *IP* concreta y el nombre del esqueleto que se ha creado. Esto se debe a que de esta forma no suceden confusiones y problemas de transmisión.

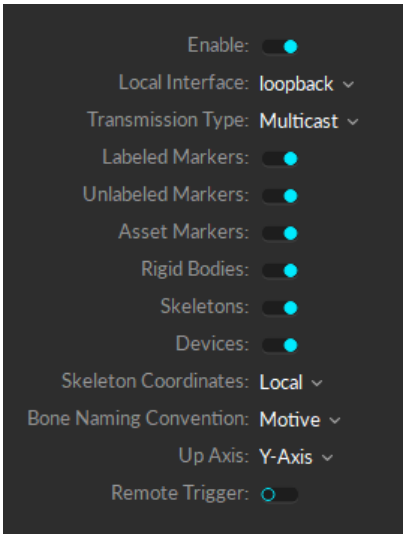

Figura 12. Configuración Data Streaming en Motive

Una vez se configure dichas opciones, Motive ya está transmitiendo la información. Lo siguiente es configurar el cliente de Unity para realizar la recepción de la información que está enviando Motive. Se recomienda una versión de Unity 2017.2/2017.1 o superior y de ***Visual Studio*** la de 2019 o superior. Para el trabajo se ha optado por la versión de Unity 2020.3.34f1, ya que ha sido con la que se ha trabajado durante el trayecto en la Universidad y es siempre la que se ha recomendado por los profesores. Además, tras varias pruebas es la que mejor capta la información transmitida por el servidor.

Se trabaja en Unity sobre el proyecto ya creado para conseguir recibir dicha información que está transmitiendo Motive. Como se ha indicado anteriormente, se va a hacer uso del *plugin* creado por Unity y OptiTrack. Al descargarlo se abre automáticamente la aplicación de Unity y se selecciona el proyecto donde se va a utilizar. Una vez se abre, en la ventana de proyecto se puede observar cómo se ha añadido una carpeta la cual incluye todos los *scripts* necesarios para poder llevar a cabo la recepción de información, además de objetos de prueba como un personaje y un objeto rígido para comprobar el funcionamiento. En la siguiente figura 13 se puede ver dicha carpeta en el proyecto.

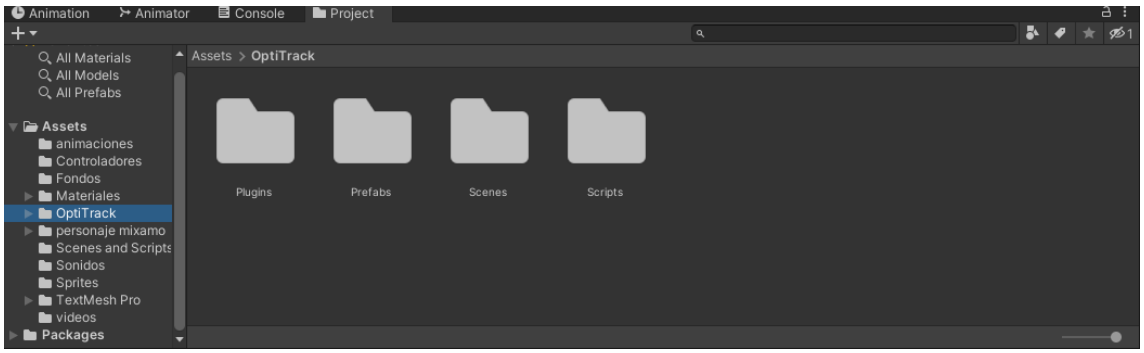

Figura 13. Carpeta OptiTrack importada en el proyecto

En el caso de este trabajo, se van a realizar las pruebas con los personajes definitivos descargados en **Mixamo** comentado anteriormente. Es necesario disponer en la escena del personaje que va a realizar a los movimientos del usuario y añadirle el *script Skeleton Animator.cs* (que viene en el *plugin* comentado anteriormente). Al añadir dicho *script* se puede observar en la ventana de ***Inspector*** que aparecen tres parámetros para rellenar, uno de los cuales es el objeto vacío con el cliente (que también viene con en el *asset* del *plugin*), otro el nombre del esqueleto que se ha creado en la aplicación de Motive y, por último, el avatar del personaje que se le va a añadir. En esta última opción, el avatar que se importa es un personaje en formato *fbx* o *blend* sin animaciones, que corresponde al avatar del personaje controlado por el usuario. En la figura 14 se muestran dichos parámetros.

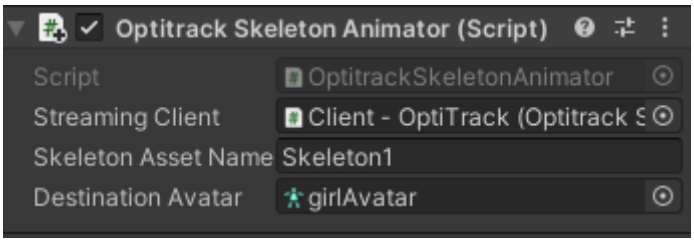

Figura 14. Script de los personajes en tiempo real

Lo último que falta para poder visualizar los movimientos en tiempo real en Unity es añadir al proyecto dicho cliente que viene configurado como un *prefab* en el *plugin* importado. Una vez añadido el cliente, en la ventana del inspector se puede observar que hay un *script* añadido automáticamente con diferentes parámetros vacíos y a falta de configurar. En el parámetro de ***Server Address*** se debe de escribir la *IP* del ordenador desde donde se manda la información; es decir, el que tiene en ejecución Motive. Y, por otro lado, la variable de ***Local Address*** corresponde a la *IP* del ordenador que está recibiendo dicha información y está ejecutando el motor de videojuegos de Unity.

Hay que tener en cuenta varias cosas a la hora de realizar la comunicación. Durante la realización del proyecto se han hecho varias pruebas y varias conexiones con diferentes ordenadores y redes. Como la aplicación de pago de Motive se encuentra en un ordenador exclusivamente configurado para usarse con la red de la universidad, ha sido imposible comunicar dos ordenadores entre sí ya que debido a los permisos y al *firewall* nunca llegaba a realizarse dicha comunicación. Pero en una red y ordenadores de uso diario el *plugin* funciona para comunicar diferentes dispositivos.

En el proyecto, se han utilizado las aplicaciones de Motive y Unity en el mismo *PC*, en concreto, en el *PC* que está conectado el sistema OptiTrack. Las *IP* que se introducen serán 127.0.0.1 que corresponden al localhost del ordenador y de esta forma no hay ningún problema para realizar la comunicación entre los dos sistemas. El resto de los parámetros se deben de dejar como viene por defecto ya que son los que recomiendan OptiTrack y Unity para realizar comunicaciones de este tipo como se puede observar en la figura 15.

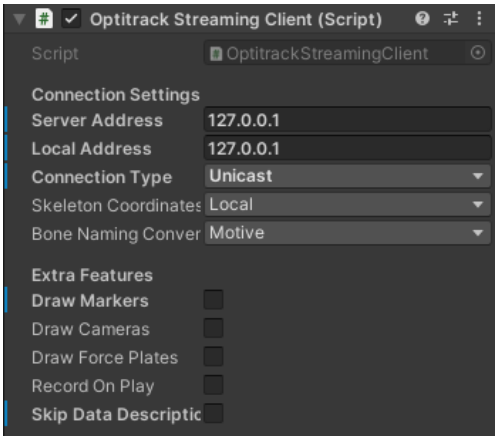

Figura 15. Parámetros Objeto Cliente

De esta forma ya está todo listo para la ejecución y para comprobarlo lo que se debe de realizar es ejecutar Unity con el botón de ***Play*** situado en la parte superior de la aplicación. En caso de que hay algún problema o algún fallo, el propio motor de videojuegos muestra un mensaje del error que está sucediendo como, por ejemplo, que el esqueleto no se está encontrando. El motivo más común durante la realización del proyecto ha sido que se introduce un nombre incorrecto que no coincide con el nombre del esqueleto en la aplicación Motive como se muestra en la siguiente figura 16.

Figura 16. Error con el nombre del esqueleto

Capítulo 4. DESARROLLO DE LA APLICACIÓN

En los siguientes apartados se va a detallar como se ha llevado a cabo el desarrollo del entorno virtual en Unity y el diseño que se ha optado para la aplicación, además de la programación interna de las interacciones del usuario y la integración de los datos de captura en tiempo real y grabadas.

4.1 Creación del entorno virtual en Unity

El desarrollo del entorno virtual en Unity es fundamental a la hora de crear una experiencia inmersiva y realista para el usuario. Al fin y al cabo, por muy precisos que sean los datos de captura de movimiento transmitidos si el entorno virtual no tiene una apariencia adecuada para el usuario lo anterior no sirve para nada. Por eso en este capítulo, se desarrollará cómo se ha elaborado dicho entorno para que los usuarios interactúen y realicen los movimientos capturados con Motive.

4.1.1 Diseño de escenarios y objetos

Antes de sumergirnos en la creación y diseño de escenarios y objetos para la aplicación, es fundamental concebir una idea y visualizar cómo plasmarla en el proyecto. Tras días de reflexión, se llega a la decisión de implementar 4 escenas clave: la pantalla de carga, la selección de personaje, la elección de ejercicio físico y, por último, una pantalla final que mostrará todos los parámetros seleccionados previamente en una sola escena. A continuación, se explica detalladamente cómo se ha llevado a cabo cada una de ellas.

Se inicia por la escena principal, que es la más crucial del proyecto y, por ende, es esencial crearla primero para realizar las pruebas necesarias. Para lograrlo, se ha diseñado una pequeña sala con suelo, paredes y techo, donde se sitúa el personaje instructor predeterminado de la aplicación sobre una plataforma. Unity ofrece herramientas que facilitan esta tarea, permitiendo crear objetos 3D que se ajusten a las características de la sala. En este caso, se ha empleado cubos deformados y un plano para actuar como suelo, lo que ha resultado en una creación sencilla y efectiva. Además, para mejorar la apariencia de los objetos creados, se incorporarán diversas texturas que realcen la claridad y permitan distinguir las diferencias entre las superficies de la sala. Este proceso se logra mediante la creación de materiales, una opción exclusiva que ofrece Unity, y aplicando dichos materiales a cada objeto correspondiente. En las figuras 17 y 18, se puede apreciar desde el entorno de Unity cómo se ve finalmente la sala tras la adición de estas texturas.

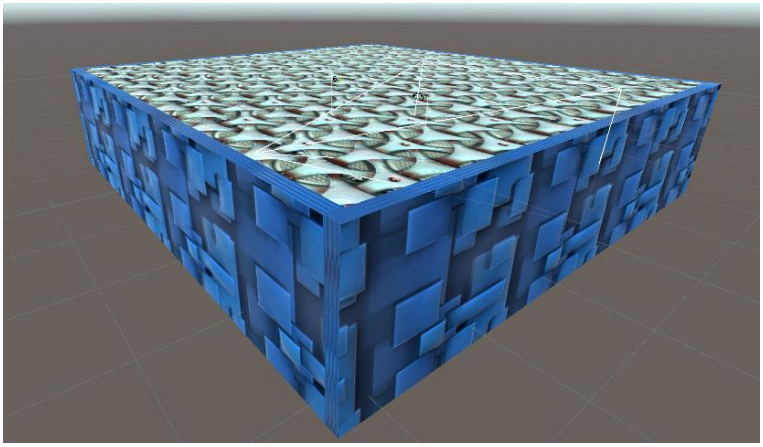

Figura 17. Vista externa de la sala

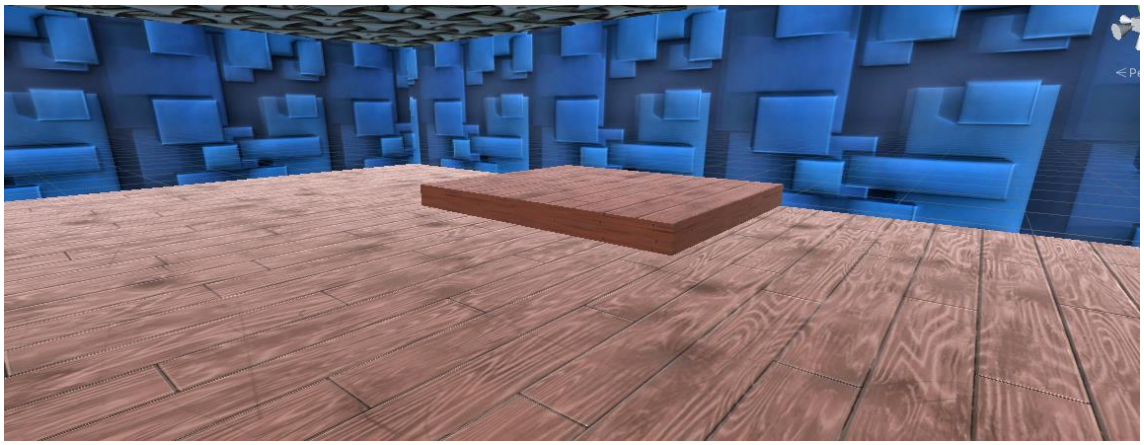

Figura 18. Vista interna de la sala

El siguiente paso en esta escena es crear al personaje que actuará como instructor de la aplicación, realizando los ejercicios seleccionados por el usuario. Se seleccionó un personaje que encaja a la perfección con el tema de la aplicación, como se puede apreciar en la figura 19.

Figura 19. Personaje Instructor

El siguiente paso consiste en determinar qué personajes estarán disponibles para que el usuario los elija en la aplicación. Para lograrlo, se ha decidido incluir un personaje femenino y otro masculino. Al igual que el personaje instructor mencionado

anteriormente, se utilizó la página web Mixamo, la cual proporciona personajes con las características apropiadas, tal como se muestra en las figuras 20 y 21.

Figura 20. Personaje masculino

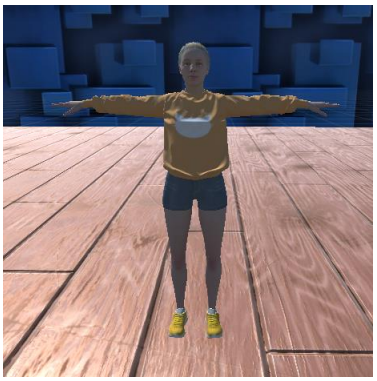

Figura 21. Personaje femenino

Una vez finalizado el diseño e integración de personajes en la escena final, el siguiente paso es crear la escena dedicada a la selección del ejercicio elegido por el usuario. Dado que existen infinitos ejercicios físicos para el entrenamiento motor, en este trabajo se han escogido 4 movimientos: Zancada, Sentadilla, *Jumping Jacks* y *Burpees*. Tanto el usuario que está utilizando la aplicación en tiempo real como el personaje instructor realizarán estos movimientos, siendo grabados para su posterior uso. En las secciones siguientes se detallará cómo se importan estas animaciones desde Motive a Unity y cómo el usuario puede interactuar con su personaje seleccionado en tiempo real, tal como se mencionó anteriormente.

Para elaborar la pantalla, se procederá a crear una nueva escena dentro de la carpeta **Escenas**. Unity permite realizar múltiples escenas en un solo proyecto, lo que facilita la organización de diferentes elementos en la aplicación o videojuego. Una vez creada la escena, se añadirá un objeto ***Canvas***, que servirá como contenedor para distribuir imágenes y texto en la pantalla. En este ***Canvas*** se colocará un título y se incluirán 4 imágenes junto a sus respectivos nombres los cuales más adelante se añadirán mediante botones para así poder seleccionarlo por el usuario. La figura 22 muestra cómo se organiza esta escena con los elementos mencionados.

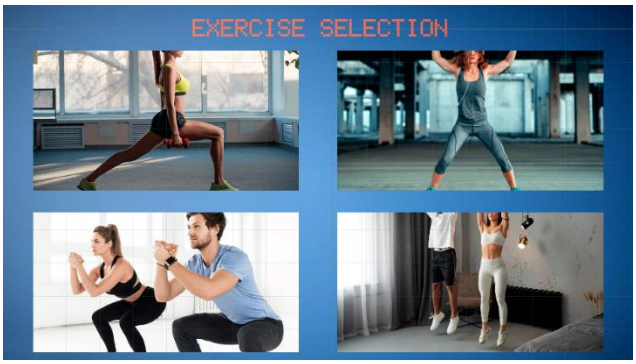

Figura 22. Escena selección de ejercicio

A continuación, se lleva a cabo el diseño de la escena de selección de personaje, utilizando otro ***Canvas*** para incluir dos imágenes representando un chico y una chica, que actuarán como botones en la programación. Además, se agrega otra imagen y un texto con un aviso que advierte que, si no se selecciona ningún personaje, se escogerá por defecto la chica. Esta imagen también funcionará como botón para iniciar la aplicación y pasar a la siguiente escena. La figura 23 muestra cómo quedará esta escena con los elementos mencionados.

Figura 23. Escena selección de personaje

Finalmente, se desarrolla la pantalla de carga de la aplicación mediante la creación de un logotipo en **Photoshop**, que hace referencia al contenido de la aplicación, y un video en **Premiere** para simular una pantalla de carga realista. El logotipo se agrega como imagen en la sección de configuración, específicamente en el apartado ***Player***. Por otro lado, para incorporar el video a la aplicación, se crea una nueva escena y se agrega un objeto vacío denominado ***Player***. Luego, utilizando el componente ***Video Player*** disponible en Unity, se incluye el video en formato *mp4*. A continuación, se muestra la imagen del logotipo en la figura 24, una captura de pantalla de la pantalla de carga en la figura 25 y una captura de cómo se vería el inspector del objeto vacío ***Player*** con el componente previamente mencionado en la figura 26.

Figura 24. Logotipo de la aplicación

Figura 25. Pantalla de carga de la aplicación

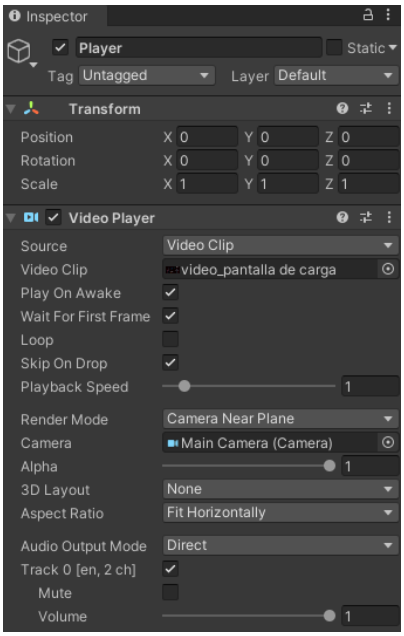

Figura 26. Inspector objeto vacío 'Player'

4.1.2 Interfaz y programación de botones

Una vez diseñada la aplicación con las 4 escenas que se utilizarán, es necesario abordar cómo el usuario puede interactuar con dichas pantallas y cambiar entre ellas, lo cual es fundamental para el desarrollo correcto del proyecto.

La explicación se desarrollará en el orden de aparición de las escenas. En la pantalla de carga, se ha implementado un *script* llamado *Pass_Scene_Video.cs*. Este código permite que, una vez finalizada la reproducción del vídeo deseado importado, la aplicación pase automáticamente a la siguiente escena. Para lograrlo, se crean dos componentes vacíos en Unity, donde se añadirán el vídeo correspondiente y el nombre de la escena a la que se desea cambiar. Las figuras 27 y 28 muestran el código del *script* y cómo se configura en el inspector de Unity, respectivamente.

```
using UnityEngine;
using UnityEngine.Video;
using UnityEngine.SceneManagement;

public class Pass_Scene_Video : MonoBehaviour
{
    public VideoPlayer videoPlayer;
    public string nextSceneName;

    private void Start()
    {
        videoPlayer.loopPointReached += OnVideoFinished;
    }

    private void OnVideoFinished(VideoPlayer player)
    {
        SceneManager.LoadScene(nextSceneName);
    }
}
```

Figura 27. Pass_Scene_Video.cs

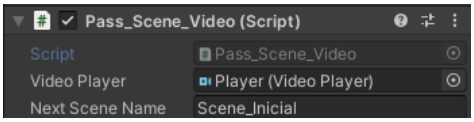

Figura 28. Inspector Player

Una vez que se ha conseguido pasar a la escena de selección de personaje, en esta etapa se incorporarán dos botones que permitirán al usuario pausar o reanudar la música de fondo durante la ejecución de la aplicación, además de un botón para salir de la aplicación. Estos botones estarán disponibles en todas las pantallas para que el usuario pueda utilizarlos en cualquier momento durante la ejecución. Además, se realizarán las configuraciones necesarias en las imágenes previamente agregadas para que funcionen adecuadamente como botones y cumplan con sus funciones respectivas.

En este proceso, se añadirán dos nuevos botones al ***Canvas*** de la escena, nombrados como *sound* y *quit*. A cada botón se le asignará un componente de imagen para darles forma. Además, se incluirá un componente vacío en el espacio de trabajo de la escena, denominado **Funciones**, el cual actuará como almacén de *scripts* para facilitar su acceso.

Para el botón de sonido se crea un nuevo *script MuteButton.cs* para que al pulsarlo cambie de activado o desactivado, además de pausar o reanudar la música. En la figura 29 se puede observar la parte principal de dicho código.

```
    void UpdateButtonSprite()
    {
        if (backgroundMusic.IsMuted)
        {
            buttonImage.sprite = musicOffSprite;
        }
        else
        {
            buttonImage.sprite = musicOnSprite;
        }
    }
}
```

Figura 29. Parte MuteButton.cs

En este momento, se crearán dos *scripts*: *Buttons.cs*, que irá aumentando su tamaño conforme se creen nuevos botones en las siguientes partes del proyecto, y *GuardarPersonaje.cs*, que almacenará la selección de personaje realizada por el usuario y permitirá pasar a las siguientes escenas manteniendo dicha opción. El *script Buttons.cs* hace referencia al botón de cerrar la aplicación, este último código puede observarse en la figura 31.

```
public void OnQuitButtonClick()
{
    #if UNITY_EDITOR
        UnityEditor.EditorApplication.isPlaying = false;
    #else
        Application.Quit();
    #endif

}
```

Figura 31. Función cerrar aplicación en Buttons.cs

Además, esta escena se completará añadiendo los componentes de botón a las imágenes de los personajes y al botón de inicio ***Start***, y se incorporará un sonido a este último junto con su respectiva función de código. Los personajes y el botón de inicio se enlazarán al *script GuardarPersonaje.cs*, tal como se muestra en la figura 32 y 33, con sus funciones correspondientes en el código. Asimismo, se creará un nuevo código para agregar el sonido deseado al botón de inicio.

Figura 32. Enlace personaje femenino al código

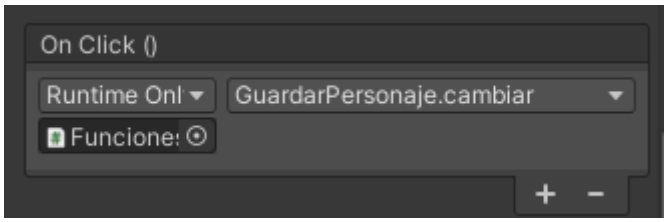

Figura 33. Enlace botón Start al código

Con esto ya se tendría configurada la escena de selección de personaje correctamente como se puede observar en la siguiente figura 34.

Figura 34. Escena selección de personaje con el personaje masculino seleccionado

A continuación, se describirá la configuración de la interfaz en la escena de selección de ejercicio. En primer lugar, se incorporan los botones de sonido y finalización de la aplicación, de manera similar a la escena anterior. Sin embargo, se añade un nuevo botón que permite al usuario regresar a la escena anterior en caso de que desee seleccionar otro personaje.

Por otro lado, para indicar los diferentes ejercicios, además de las imágenes que se habían introducido anteriormente, se añaden 4 botones donde aparecen los nombres de estos. Estos botones mandan al usuario a la escena final con el personaje escogido anteriormente, además de que el personaje instructor realiza el ejercicio correspondiente. El tema de las animaciones en el personaje instructor se abordará más adelante. Para la configuración de los botones se añadirá en el *script Buttons.cs* sus respectivas líneas de código de la siguiente forma como se puede observar en la figura 35.

```
public void OnBurpeeButtonClick()
{
    // Almacenar la selección del ejercicio en PlayerPrefs
    PlayerPrefs.SetString("SelectedExercise", "Burpee");

    // Cargar la siguiente escena
    SceneManager.LoadScene("Scene_principal");
}
```

Figura 35. Función botón para el caso de Burpee

Por tanto, la escena de selección de ejercicio quedaría de la siguiente forma como se puede observar en la figura 36.

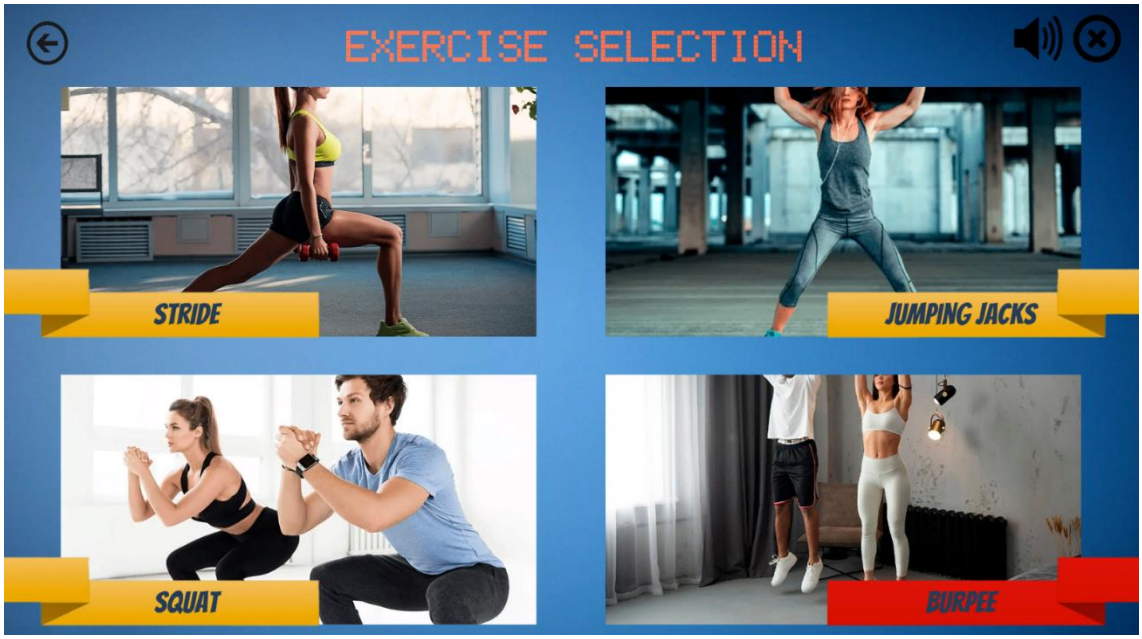

Figura 36. Escena selección de ejercicio con el Burpee seleccionado

En la fase final del desarrollo de la aplicación, se procede a configurar la escena final. Los botones de sonido y finalización de la aplicación se mantienen, pero se reemplaza el botón de flecha por uno que reinicia el ejercicio, en caso de que el usuario desee volver a empezar. Además, se agregan dos botones nuevos debajo de la escena, que permiten al usuario cambiar de personaje o seleccionar un nuevo ejercicio.

Sin embargo, al iniciar la escena final, surgen dos problemas en la aplicación. El primero es que, a pesar de seleccionar el personaje deseado por el usuario en la primera escena, ambos personajes aparecen sobreexpuestos, como se muestra en la figura 37.

Figura 37. Error personajes sobreexpuestos

Para resolver esto, es necesario deseleccionar ambos personajes en el inspector y crear un nuevo *script* llamado *CargarPersonaje.cs*. Este *script* mostrará el personaje seleccionado por el usuario en la primera pantalla de la aplicación de forma correcta. El código toma la variable del personaje almacenada en el *script GuardarPersonaje.cs* y, según la selección, destruye el otro personaje virtual. La figura 38 muestra la parte esencial de dicho código.

```
private void Update()
{
    girl = PlayerPrefs.GetInt("girlSelect") == 1;     // lee el personaje que esta seleccionado
    boy = PlayerPrefs.GetInt("boySelect") == 1;


    if(girl == true)
    {
        girlPersonaje.SetActive(true);          // cuando lee  que un personaje esta guardado el otro lo destruye
        Destroy(boyPersonaje);
    }

    if(boy == true)
    {
        boyPersonaje.SetActive(true);
        Destroy(girlPersonaje);
    }
}
```

Figura 38. Parte del código CargarPersonaje.cs

El segundo problema, es que al pasar a la pantalla final la escena se inicia inmediatamente y al usuario no le da tiempo a prepararse ni sabe cuándo va a iniciar el ejercicio. Para ello, se ha diseñado un contador de cuenta atrás de 5 segundos para que el usuario sepa cuándo comenzará el ejercicio. Este contador se mostrará en la pantalla de la escena y para implementarlo, se crea un nuevo *script* denominado *CountdownTimer.cs*. El *script* utiliza una variable de segundos que va disminuyendo hasta alcanzar cero, momento en el cual el objeto se destruye y la escena se inicia. El código se agrega a un nuevo objeto de tipo texto importado en el espacio de trabajo de esta escena. La figura 39 muestra el código correspondiente a este contador.

```
public class CountdownTimer : MonoBehaviour
{
    public Text countdownText; // Referencia al componente Text

    private void Start()
    {
        // Iniciar la cuenta regresiva
        StartCoroutine(StartCountdown());
    }

    private IEnumerator StartCountdown()
    {
        int currentTime = 5;

        // Actualizar el texto inicial
        UpdateCountdownText(currentTime);

        while (currentTime > 0)
        {
            yield return new WaitForSeconds(1f); // Esperar 1 segundo

            currentTime--; // Restar 1 al tiempo actual
            UpdateCountdownText(currentTime); // Actualizar el texto
        }

        // La cuenta regresiva ha finalizado, puedes ejecutar alguna acción adicional aquí

        countdownText.text = "¡GO!"; // Actualizar el texto a "Fin" o lo que desees mostrar al finalizar
        Destroy(gameObject,2f);
    }

    private void UpdateCountdownText(int time)
    {
        countdownText.text = time.ToString(); // Actualizar el texto con el tiempo actual
    }
```

Figura 39. CountdownTimer.cs

Posteriormente, se realizan ajustes en la cámara para resolver un problema en la escena donde el personaje del usuario desaparece, es decir, que el personaje no está en el campo de visión de la cámara. Se utiliza la función ***Cinemachine*** de Unity para crear una nueva componente en la cámara y asegurarse de que siga al personaje principal durante la escena. La figura 40 muestra este cambio en el inspector de la cámara.

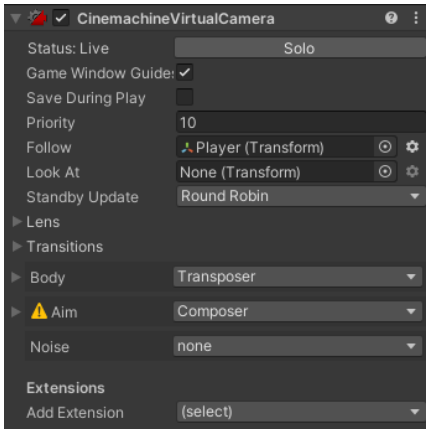

Figura 40. CinemachineVirtualCamera

Finalmente, para indicar al usuario que la aplicación ha terminado, se crea un nuevo ***Canvas*** que aparece después de 10 segundos de ejecución del ejercicio (este tiempo puede ajustarse en el inspector). Este ***Canvas*** muestra una nueva pantalla donde el usuario puede volver a la pantalla de selección de personaje, a la pantalla de selección de ejercicio o cerrar la aplicación. Para lograr esto, se ha desarrollado un **script** llamado *ExerciseManager.cs*, que maneja esta función. Además, más adelante en este mismo *script* se agregarán los controles de las animaciones del personaje instructor. Esto se debe a que se desea pasar a la segunda pantalla una vez que la animación haya finalizado, por lo que ambas funciones se han incorporado en un mismo *script*. La figura 41 muestra la función que activa el segundo ***Canvas***.

```
// Activar el segundo canvas
secondCanvas.SetActive(true);
```

Figura 41. Función para activar el segundo Canvas

Así, el diseño de la interfaz de la última escena de la aplicación, con ambas pantallas (la principal y la de finalización), ha quedado configurado. En las figuras 42 y 43 se puede apreciar cómo lucen estas pantallas.

Figura 42. Pantalla final

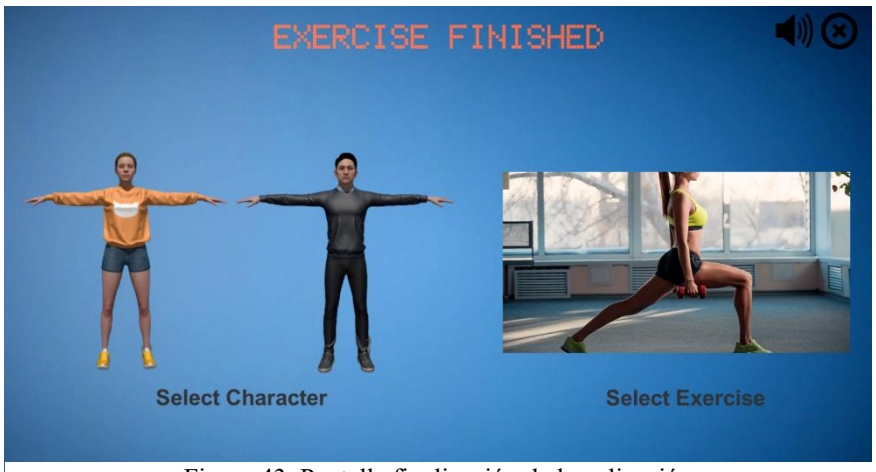

Figura 43. Pantalla finalización de la aplicación

4.2 Integración de los datos de captura

En los próximos apartados, se describirá la implementación central de este proyecto: la integración de los datos de captura en tiempo real y grabados desde el *software* OptiTrack Motive al motor de videojuegos Unity, específicamente en los personajes previamente integrados en la aplicación, tal como se ha mencionado en secciones anteriores. Para lograrlo, se abordarán temas como la recepción y procesamiento de los datos capturados en tiempo real, así como los datos previamente grabados, y se describirá cómo se asignan estos movimientos a los personajes virtuales.

4.2.1 Asignación de movimientos a personajes virtuales

Este aspecto es fundamental para el desarrollo coherente de la aplicación, ya que aborda cómo se asignan los movimientos capturados en tiempo real y grabados a los personajes virtuales, incluido el personaje instructor y los personajes predefinidos para el usuario final.

Inicialmente, para vincular los movimientos en tiempo real del usuario final a los personajes virtuales de la aplicación, se requiere una colocación precisa del traje de captura en el cuerpo. En el caso específico del esqueleto ***Baseline 37***, es imperativo que todos los marcadores se ubiquen según las pautas proporcionadas por la aplicación Motive. Esta disposición es crítica, ya que cualquier desviación puede afectar la captura precisa del movimiento, lo que a su vez comprometería el rendimiento de la aplicación.

Una vez que el traje y los marcadores están correctamente alineados, se procede a revisar nuevamente la configuración de la transmisión, tanto en Motive como en el *plugin* de Unity, según lo explicado en apartados anteriores. Con estas medidas en su lugar, al ejecutar la aplicación, se puede observar en la pantalla final al personaje seleccionado replicando fielmente los movimientos en tiempo real del usuario con el traje. Esto indica que la asignación de los movimientos en tiempo real se ha ejecutado correctamente y que la sincronización entre la captura de movimiento y la animación virtual ha sido exitosa como se puede observar en la figura 44.

Figura 44. Correcto funcionamiento en tiempo real

Además, en relación con la captura de movimiento del personaje instructor, se procede a registrar los movimientos de los 4 ejercicios elegidos para este proyecto. Esta etapa implica aprovechar la funcionalidad de grabación de movimiento proporcionada por el *software* Motive. Esta opción se ubica en la parte inferior de la pantalla en el modo de captura, presentada como un botón de color rojo como se puede observar en la figura 45. Es fundamental destacar que esta operación se realiza considerando la previa creación del esqueleto ***Baseline 37***.

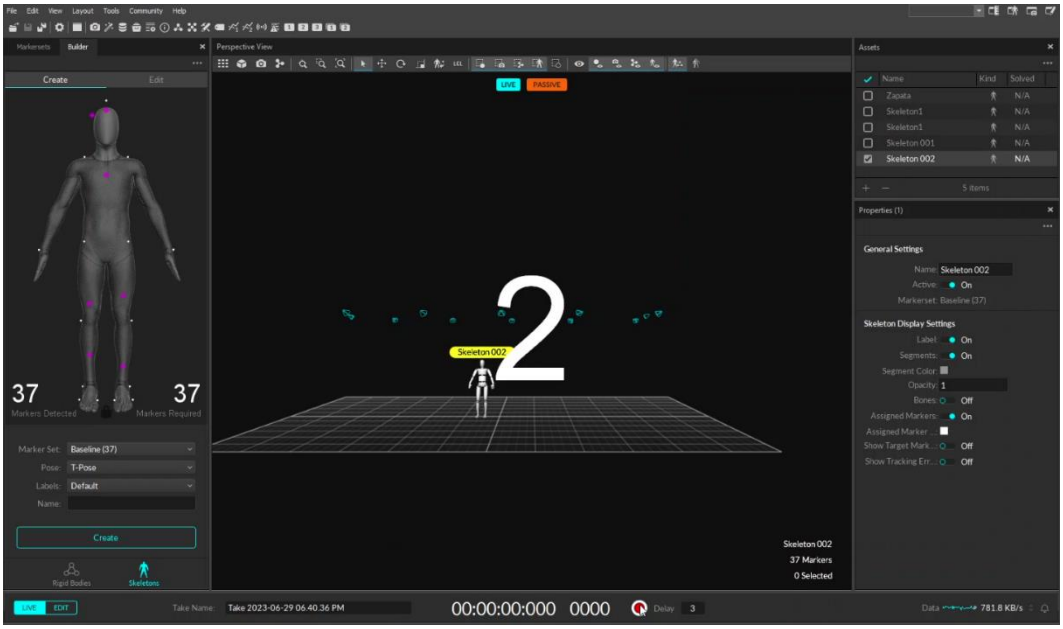

Figura 45. Modo grabación en Motive

Una vez completada la grabación de un movimiento, este se guarda en la carpeta del proyecto, claramente visible en la interfaz de edición del *software*, con su nombre correspondiente. Con este paso finalizado, el movimiento queda registrado con éxito mediante OptiTrack. Posteriormente, para extraer y llevar estos movimientos a Unity, simplemente se requiere exportar el archivo en formato *fbx*, que como se ha mencionado previamente, es una de las extensiones altamente compatible con Unity para la importación de objetos, animaciones y efectos. Este proceso se realiza de manera individual para cada uno de los ejercicios presentes en la aplicación.

A continuación, surge un desafío relacionado con cómo asignar al personaje instructor el movimiento adecuado de acuerdo con el ejercicio seleccionado por el usuario en la pantalla previa. Inicialmente, se consideró duplicar la escena final en cuatro versiones y que, en función del botón presionado, el usuario fuera redirigido a la escena correspondiente a cada ejercicio. No obstante, se concluyó que esta no era una opción eficiente. Tras un proceso de reflexión y numerosas pruebas, se optó por una solución diferente. En una única escena, mediante las capacidades del ***Animator*** de Unity, se logra que el personaje realice el movimiento apropiado en función de la selección previa realizada por el usuario [12].

Para ello en el *script Buttons.cs* se añade las líneas de código que corresponden a la pantalla de selección de ejercicio, donde en este código según el ejercicio que se escoja, se guardará esa variable para que así en la pantalla final se realice el ejercicio

correspondiente. En la figura 46 se puede observar un ejemplo de selección de uno de los ejercicios en dicho código.

```
public void OnBurpeeButtonClick()
{
    // Almacenar la selección del ejercicio en PlayerPrefs
    PlayerPrefs.SetString("SelectedExercise", "Burpee");

    // Cargar la siguiente escena
    SceneManager.LoadScene("Scene_principal");
}
```

Figura 46. Selección de ejercicio Burpee

Después de incorporar las líneas de código necesarias, el siguiente paso implica agregar las instrucciones adecuadas para controlar la animación del personaje instructor. Tal como se mencionó con antelación, en el *script ExerciseManager.cs*, se introducen las líneas correspondientes a las acciones del ***Animator***. Para ello, se importa la variable creada en el código previo, como se muestra en la figura 47, y creamos una estructura condicional utilizando la instrucción *if*, en la que el movimiento del instructor se determina según el ejercicio seleccionado. Este proceso se ejemplifica en la figura 48.

```
// Obtener el ejercicio seleccionado de PlayerPrefs
string selectedExercise = PlayerPrefs.GetString("SelectedExercise");
```

Figura 47. Importar variable con PlayerPrefs

```
if (selectedExercise == "Stride")
{
    animator.SetBool("Stride", true);
    animator.SetBool("Squat", false);
    animator.SetBool("Burpee", false);
    animator.SetBool("Jumping Jacks", false);
}
```

Figura 48. Condicional ejercicio de Stride

De esta forma ya se ha realizado la programación de la asignación de movimiento a los personajes de la escena final de la aplicación.

4.2.2 Almacenamiento de Movimientos en Tiempo Real

El almacenamiento de movimientos en tiempo real del usuario final en la aplicación es una parte esencial de la interacción entre el mundo virtual y el mundo real. Este proceso se logra gracias a la integración del *software* de OptiTrack Motive, que permite capturar y guardar los movimientos realizados por el usuario mientras utiliza la aplicación. Para lograrlo, como se ha explicado anteriormente, el usuario lleva puesto un traje equipado con sensores, los cuales son rastreados por el sistema de OptiTrack en tiempo real.

Durante la ejecución de un ejercicio en la aplicación, mientras el usuario realiza los movimientos correspondientes, el *software* Motive registra y rastrea la posición y orientación de los marcadores en el traje del usuario. Simultáneamente, estos movimientos son reflejados en el mundo virtual de la aplicación, lo que permite una experiencia de inmersión y realismo. Un aspecto crucial es que, durante la grabación en tiempo real, la ventana de captura de Motive está activa y registrando continuamente los datos de movimiento del usuario. Una vez que el usuario finaliza el ejercicio y detiene la grabación en Motive, los datos capturados se almacenan en un archivo específico en la ventana de edición del *software*.

La versatilidad del almacenamiento de los datos capturados radica en la posibilidad de exportarlos en diferentes formatos. En el trabajo se ha optado por exportarlos en formato *fbx* y se puede usar el archivo posteriormente para evaluar la correcta realización del ejercicio. Este formato es ampliamente utilizado en el ámbito de la animación y los videojuegos, ya que es compatible con diversas plataformas y *software*. Al exportar el archivo en formato *fbx*, este puede ser utilizado por el entrenador o fisioterapeuta para comprobar la ejecución del ejercicio. Este proceso no solo asegura una experiencia más inmersiva y personalizada para el usuario, sino que también amplía las posibilidades creativas al permitir que los movimientos del mundo real se traduzcan directamente en el comportamiento de los personajes virtuales en la aplicación.

Capítulo 5. RESULTADOS Y EVALUACIÓN

Este segmento del proyecto se enfoca en la evaluación de los logros alcanzados durante la creación de la aplicación, incluyendo la validación y pruebas del *software*, así como el análisis de los resultados obtenidos.

5.1 Validación y pruebas de la aplicación

La validación y las pruebas de la aplicación representaron etapas fundamentales en la búsqueda de un trabajo de calidad. Estas fases abarcaron una serie de evaluaciones destinadas a explorar y validar distintos aspectos de la aplicación en términos de funcionamiento, usabilidad y precisión en la captura y reproducción de movimientos. La integración de estos factores garantizó una experiencia más enriquecedora y ajustada a las necesidades del usuario. Para examinar la precisión de la captura de movimientos en tiempo real, se llevaron a cabo una variedad de ejercicios que abarcaban desde movimientos simples hasta gestos más complejos, con el objetivo de verificar la exactitud entre los movimientos reales y su representación en el entorno virtual de OptiTrack. Estas pruebas permitieron afinar la configuración de la captura y ajustar la precisión de los marcadores, contribuyendo así a una captura más fiable y precisa de los movimientos del usuario.

En paralelo, las pruebas de usabilidad se centraron en evaluar la experiencia del usuario al interactuar con la interfaz de la aplicación. Los participantes llevaron a cabo tareas específicas, como seleccionar personajes y ejercicios, mientras se registraban sus acciones y comentarios. Esto proporcionó información valiosa sobre la navegación y la comprensión de las instrucciones. Los comentarios de los usuarios guiaron las mejoras en la interfaz, lo que resultó en una experiencia más intuitiva y agradable. Es importante resaltar que estas pruebas no solo se centraron en la funcionalidad, sino también en la experiencia del usuario. Los comentarios y las observaciones de los usuarios enriquecieron la comprensión de cómo interactuaban con la aplicación y qué aspectos requerían atención adicional. Las actualizaciones contribuyeron significativamente a la creación de una aplicación más eficaz y satisfactoria.

5.2 Análisis de los Resultados Obtenidos

El análisis exhaustivo de los resultados se enfocó en tres aspectos centrales: la precisión de la captura de movimientos en tiempo real, la experiencia del usuario en la interfaz y la eficacia en la incorporación de los movimientos capturados en los personajes virtuales de la aplicación.

En referencia a la captura de movimiento en tiempo real, los resultados obtenidos demostraron una fidelidad positiva entre los movimientos reales ejecutados por el usuario

y los que se capturaron en tiempo real en el entorno virtual. A pesar de que surgieron desafíos en la fase de calibración, tanto en la disposición del espacio como en la colocación precisa del traje y los marcadores con respecto al esqueleto ***Baseline 37***, estos obstáculos se superaron con éxito, permitiendo una captura de movimientos coherente y fiable.

En relación con la respuesta del usuario a la interfaz, los datos recopilados reflejaron que los participantes experimentaron fluidez en la interacción con la aplicación. Los resultados sugieren que las actualizaciones iterativas realizadas en función de la retroalimentación recibida desempeñaron un papel crucial en la creación de una experiencia más intuitiva y amigable. El proceso de refinamiento guiado por la retroalimentación de los usuarios contribuyó significativamente a la funcionalidad y facilidad de uso de la aplicación.

En lo que respecta a la incorporación de los movimientos capturados en los personajes virtuales, los resultados fueron satisfactorios. La utilización de la extensión *fbx* en los archivos exportados permitió una transferencia exitosa de los movimientos capturados desde el *software* Motive al motor de Unity además de la ayuda del *plugin* creado por ambas empresas. La comunicación fluida entre estos dos *softwares* permitió una integración sin inconvenientes, asegurando que los movimientos se reprodujeran de manera coherente y realista en los personajes virtuales de la aplicación.

En conclusión, el análisis de los resultados reveló que la aplicación alcanzó los objetivos planteados en términos de precisión en la captura de movimientos en tiempo real, interacción del usuario con la interfaz y efectividad en la importación de movimientos en los personajes virtuales. Estos hallazgos no solo validan el enfoque y las decisiones tomadas en el desarrollo, sino que también apuntan a áreas de mejora futuras para seguir elevando la calidad y la experiencia del usuario en futuras iteraciones del proyecto.

Capítulo 6. DISCUSIÓN Y CONCLUSIONES

Concluyendo, en las secciones a continuación se procederá a evaluar si los objetivos delineados al inicio del desarrollo han sido alcanzados, a detallar las contribuciones y limitaciones de este trabajo, y, por último, a explorar las posibles direcciones que los futuros trabajos pueden llegar a tomar después de desarrollar la aplicación.

6.1 Cumplimiento de los objetivos planteados

Después de un análisis extenso, se puede concluir de manera sólida que los objetivos trazados al inicio del proceso de desarrollo de la aplicación han sido alcanzados de manera exitosa. El eje central del proyecto residía en la creación de una aplicación capaz de capturar el movimiento en tiempo real de un individuo mediante la avanzada tecnología de OptiTrack, para luego transferir esa dinámica al ámbito virtual. Adicionalmente, la aplicación se basa como una herramienta que empodera a los usuarios a realizar ejercicios físicos, permitiéndoles observar cómo sus acciones se traducen en movimientos de un personaje virtual, y permitiéndoles perfeccionar estos movimientos en caso de ser necesario.

Otro punto de enfoque fundamental de este trabajo se centraba en brindar a los usuarios una experiencia inmersiva y envolvente. La interfaz de la aplicación ha sido cuidadosamente diseñada con este propósito, proporcionando un camino intuitivo para seleccionar personajes, ejercicios y proporcionando un espacio claro para la realización de los movimientos elegidos. Las pruebas de usabilidad realizadas hasta ahora respaldan firmemente estas afirmaciones, confirmando que los objetivos relativos a la experiencia del usuario se han logrado con éxito.

Por último, es imperativo destacar el desafío de lograr la comunicación eficiente entre dos sistemas heterogéneos como Motive y Unity. A través de una investigación minuciosa y gracias a la implementación de un *plugin* creado conjuntamente por las empresas OptiTrack y Unity, se ha conseguido satisfacer este objetivo de manera notable, permitiendo una comunicación fluida y precisa entre las dos plataformas. En resumen, el análisis concluye que los objetivos planteados se han materializado de manera efectiva y coherente en este proyecto.

6.2 Trabajos futuros

Una perspectiva futura que se consideró al finalizar la implementación de la aplicación involucra proporcionar al usuario, al concluir la ejecución del ejercicio seleccionado, una pantalla final que no solo albergue las mismas opciones mencionadas previamente, sino también un botón que permita guardar el archivo en formato *fbx*. Esto permitiría que el proceso de grabado y exportación del movimiento se realice directamente desde la propia

aplicación, eliminando así la necesidad de transitar entre la aplicación y el *software* OptiTrack Motive para completar esta tarea.

Además, se plantea la posibilidad de introducir una característica adicional al sistema. En este escenario hipotético, se añadiría un componente de evaluación que brindaría al usuario un indicador de precisión tras completar el movimiento. Este indicador podría ser representado mediante un porcentaje, y su cálculo se basaría en una comparación entre la ejecución del usuario y la ejecución del personaje instructor o el modelo elegido por el usuario. Esta característica ampliaría el aspecto interactivo y de retroalimentación de la aplicación, brindando al usuario información más detallada sobre su desempeño y, en última instancia, promoviendo una mejora continua de la ejecución de los movimientos.

Finalmente, se podría dedicar un esfuerzo considerable a la refinación de los detalles de la interfaz, con el objetivo de alcanzar un nivel de profesionalismo aún más elevado. Este enfoque apuntaría a transformar la aplicación en una propuesta más comercial, accesible incluso para usuarios con menos familiaridad técnica. La mejora de la interfaz no solo contribuiría a la estética, sino también a la usabilidad general de la aplicación, garantizando una experiencia fluida y amigable para una audiencia más amplia.

Este proceso implicaría la revisión y optimización de la disposición de elementos visuales, la claridad de los textos y las instrucciones, así como la implementación de una navegación intuitiva. Además, se podría explorar la integración de guías visuales o tutoriales interactivos que orienten a los usuarios en su interacción con la aplicación, eliminando cualquier posible barrera que pudiera surgir debido a la falta de experiencia técnica.

Bibliografía

[1] Adobe, "Adobe," [Online]. Available: https://www.adobe.com/es/creativecloud/animation/discover/motion-capture.html.

[2] Teseo, "Teseo.es," 11 02 2020. [Online]. Available: https://teseo.es/noticias/que-es-y-como-funciona-la-captura-de-movimiento/.

[3] S. CO', "streetcommunication.com," [Online]. Available: https://streetcommunication.com/es/conjuntos-de-producci%C3%B3n-virtual/optitrack/.

[4] OptiTrack, "OptiTrack," [Online]. Available: https://docs.optitrack.com/.

[5] Wikipedia, "Wikipedia," [Online]. Available: https://es.wikipedia.org/wiki/Unity_(motor_de_videojuego).

[6] Macondo, "Unity Support," 2022. [Online]. Available: https://support.unity.com/hc/es/articles/7642130833812-Unity-Qu%C3%A9-es-y-c%C3%B3mo-funciona-.

[7] "Mastery Coding," 06 07 2021. [Online]. Available: https://www.masterycoding.com/blog/unity-best-beginner-engine.

[8] "OptiTrack," [Online]. Available: https://optitrack.com/accessories/.

[9] Adobe, "Mixamo," 2008. [Online]. Available: https://www.mixamo.com/#/.

[10] M. E. L. @KISD, "Youtube," 2021. [Online]. Available: https://www.youtube.com/watch?v=miK3YqNsskE.

[11] OptiTrack, "OptiTrack v3.0," [Online]. Available: https://docs.optitrack.com/plugins/optitrack-unity-plugin.

[12] D. Pachi, "Youtube," 2019. [Online]. Available: https://www.youtube.com/watch?v=Ay_oy6GXC-s&t=1621s.