Trabajo de Fin de Grado

Grado en Ingeniería Informática

Realidad Virtual en San Cristóbal de La Laguna Patrimonio Histórico

Virtual Reality in San Cristóbal de La Laguna Historical Heritage

Kevin Estévez Expósito

La Laguna, 3 de septiembre de 2018

D.ª **Isabel Sánchez Berriel**, con N.I.F. 42.885.838-S profesora Contratada Doctora adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

D. **Fernando Andrés Pérez Nava**, con N.I.F. 42.091.420-V profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

C E R T I F I C A N

Que la presente memoria titulada:

"Realidad Virtual en San Cristóbal de La Laguna Patrimonio Histórico"

ha sido realizada bajo su dirección por D. **Kevin Estévez Expósito**,

con N.I.F. 43.838.692-W.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 3 de septiembre de 2018

Agradecimientos

A mi tutora D.ª Isabel y a mi cotutor D. Fernando, por darme todo su apoyo, consejos y experiencia en este campo, siempre dispuestos a resolver mis dudas.

A la Universidad de La Laguna por prestarme el equipo necesario, sin el que el desarrollo habría sido mucho más lento y tedioso.

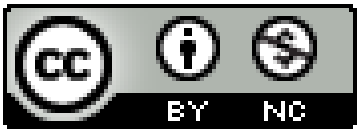
Al Excmo. Ayuntamiento de San Cristóbal de La Laguna, especialmente a D.ª Carmen Rosa, y a la Memoria Digital de Canarias, por la colaboración prestada y el suministro de datos, información y archivos históricos siempre que me ha sido necesario.

A mis profesoras y profesores por todo lo aprendido durante estos años, que me ha hecho crecer tanto como persona como profesional.

A mis amigas y amigos, en especial a Alberto por compartir conmigo su experiencia y mantener siempre vivo mi interés por el 3D y la tecnología en general.

Por último, agradecer a mi madre, a mi padre y a mis hermanos por sus consejos y su apoyo en cada una de las decisiones que, durante estos años de carrera, me han hecho llegar hasta aquí.

Licencia



Resumen

El objetivo principal de este Trabajo de Fin de Grado ha sido el desarrollo de una aplicación móvil de Realidad Virtual que pretende simular la Plaza del Adelantado de San Cristóbal de La Laguna y las edificaciones históricas colindantes. Todo ello ambientado e inspirado en el plano elaborado a finales del siglo XVI por Leonardo Torriani. De esta forma, se pretende dotar al Excmo. Ayuntamiento de San Cristóbal de La Laguna de un prototipo funcional que sirva de apoyo en sus actividades de difusión cultural y de turismo en una ciudad Patrimonio Histórico de la Humanidad como lo es San Cristóbal de La Laguna.

La aplicación se ha desarrollado para su funcionamiento en un dispositivo Samsung Gear VR con su respectivo mando controlador, motorizado por Oculus. Se ha utilizado el motor de videojuegos Unity 3D para el desarrollo de la aplicación y sus mecánicas, aprovechando la flexibilidad que esta herramienta brinda en cuanto a plataformas de desarrollo, así como la gran cantidad de activos y complementos de uso libre que se encuentran en el Asset Store del propio Unity 3D.

Palabras clave: *Unity 3D, C#, Samsung Gear VR, Oculus, Head-Mounted Display, San Cristóbal de La Laguna, Patrimonio Histórico, Plaza del Adelantado.*

Abstract

The main objective of this Final Degree Project has been the development of a mobile and Augmented Reallity application that pretends to simulate the Adelantado's Square on San Cristóbal de La Laguna and the adjacent historic buildings. All set and inspired in the map drawn up at the end of the 16th century by Leonardo Torriani. In this way, it is intended to endow the City Hall of San Cristóbal de La Laguna of a functional prototype that could be a support in its activities of cultural diffusion and tourism in a Historical Heritage city such as San Cristóbal de La Laguna.

The application has been developed to be operative on a Samsung Gear VR device with its respective controller, powered by Oculus. The Unity 3D videogame engine has been used for the development of the application and its mechanics, taking advantage of the flexibility that this tool provides in terms of development platforms, as well as the huge number of free to use assets and accessories that can be found in the Unity 3D Asset Store.

Keywords: *Unity 3D, C#, Samsung Gear VR, Oculus, Head-Mounted Display, San Cristóbal de La Laguna, Historical Heritage, Adelantado's Square.*

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

Los vaivenes históricos de la ciudad de San Cristóbal de La Laguna no han producido cambios sustantivos en sus edificaciones ni en su trazado urbano. Hacia el año 1500, el Cabildo acuerda trazar un plano de ensanche en dirección sur, estableciendo una cuadrícula (plano en damero) de calles ordenadas según el gusto que imperaba en la Europa renacentista en ese momento. De este modo, a finales del siglo XVI, el casco histórico de la ciudad quedó configurado definitivamente tal y como lo conocemos en la actualidad, como se puede observar en el primer plano que se conserva de la ciudad, realizado en 1588 por el ingeniero militar Leonardo Torriani, enviado a las islas por Felipe II para realizar la mejora de la defensa y descripción de las mismas, el cual se puede ver en la Figura 1.1 (imagen cedida por la MDC (Memoria Digital de Canarias) [1]). La coexistencia entre los viejos caserones de tradición mudéjar, los antiguos conventos, la arquitectura ecléctica, y los nuevos planes de ordenación urbana, caracterizan la etapa más contemporánea; y motivaron que el 2 de diciembre del año 1999, en Marrakech, el Comité de Patrimonio Histórico de la UNESCO, reconociera los valores singulares y de autenticidad de San Cristóbal de La Laguna, y le otorgará el título de Bien Cultural Patrimonio de la Humanidad.

Con este Trabajo de Fin de Grado se pretende dotar al Excmo. Ayuntamiento de San Cristóbal de La Laguna de un prototipo funcional de herramienta de apoyo en sus actividades de difusión cultural y de turismo en una ciudad Patrimonio Histórico de la Humanidad como lo es San Cristóbal de La Laguna. Para ello se ha hecho uso de la Realidad Virtual como herramienta para la difusión de la riqueza patrimonial de la ciudad, herramienta que, como se verá a continuación, tiene un enorme valor y potencial en el momento en el que se encuentra la sociedad actual.

Figura 1.1: Primer plano de San Cristóbal de La Laguna realizado por Leonardo Torriani [1]

1.1 Antecedentes y estado actual

En los últimos años, el concepto de la Realidad Virtual (RV) ha ido ganando terreno a pasos agigantados. Las grandes compañías tecnológicas ya han desarrollado y comercializado diversas herramientas que, poco a poco, van cambiando la experiencia de uso de nuestros dispositivos. El "boom" de este fenómeno se produce en el año 2012 con la fiebre del *Oculus Rift*, rompiendo los récords de financiación en la plataforma de crowdfunding *Kickstarter*. En la Figura 1.2 [2] se aprecia el crecimiento de patentes de sistemas HMD (*Head-Mounted Display*).

Por lo general, este tipo de dispositivos se había mantenido fuera del alcance de la mayoría del público debido a los altos costes que suponía adquirir un equipo de este tipo que tuviera unas características aceptables, además de la necesidad de disponer de un ordenador para dar soporte a dicha herramienta.

Por suerte para el consumidor, con el tiempo se han ido sumando empresas como *Samsung* con sus *Gear VR* o *Google* con sus *Cardboard* que, con un coste bastante más asequible, logran rendimientos más que aceptables gracias a sus gafas y un dispositivo móvil compatible con RV como el que la mayoría de nosotros tenemos. Esta tendencia a crear aplicaciones móviles de RV ha

propiciado que herramientas como *Unity 3D* permitan a los desarrolladores crear este tipo de aplicaciones de manera cada vez más sencilla. En concreto, *Unity 3D* permite la creación de juegos y aplicaciones de RV para dispositivos móviles Android con el SDK de *Google*, con el que el propio motor de *Unity 3D* construye el paquete de instalación de la aplicación con soporte para RV, siendo este un proceso muy cómodo para el desarrollador, que apenas debe preocuparse por ello.

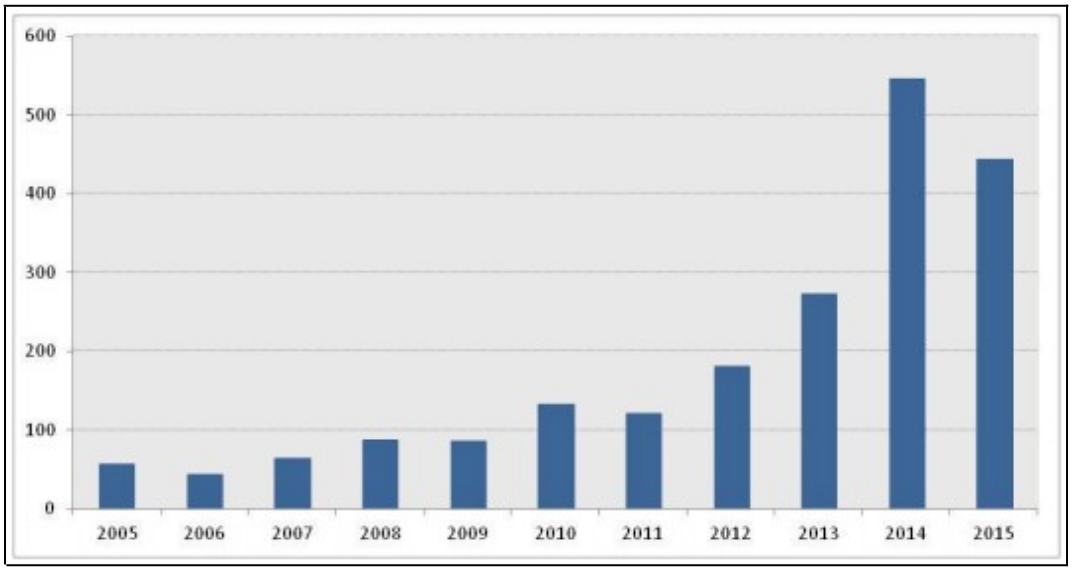


Figura 1.2: Evolución temporal del nº de patentes con keywords "Head-Mounted Display" [2]

Tanto la Realidad Virtual como la Realidad Aumentada (RA) son tecnologías en continuo crecimiento y con un gran potencial en los ámbitos que nos ocupan en el presente proyecto, como lo son el ocio y el turismo. Por una parte, la RV nos permite visitar virtualmente un posible destino antes incluso de realizar la reserva, descubriendo así el destino antes de viajar a él. Por otra parte, la RA nos permite ampliar la información que nos proporciona el mundo real, normalmente cuando nos encontramos físicamente en el destino. Ambas tecnologías pueden y suelen estar gamificadas, haciendo la experiencia más amena y divertida para el usuario.

San Cristóbal de La Laguna posee una serie de aplicaciones móviles enfocadas a la difusión de la cultura y al turismo, dentro de las que se podrían destacar la app *Turismo La Laguna* [3], que sirve al usuario de guía para ubicar y dar a conocer los diferentes elementos de la ciudad que conforman el patrimonio histórico, lugares de ocio, actividades culturales, etc, dando información útil y mostrando imágenes representativas de estos. Sin

embargo, esta ciudad no dispone de herramientas con las características del proyecto que se ha desarrollado, mientras que otras ciudades españolas ya se han unido a esta creciente tendencia, como Barcelona con su proyecto *VR Barcelona in Gothic* ya en 2011, el cual permitía sobrevolar la ciudad gótica de la época o pasear por sus calles [4]; Salamanca con la RA ofrecida por la aplicación móvil *Layar*, que permite visualizar información digital de forma interactiva [5]; Galicia con su propia aplicación móvil que, entre otras funcionalidades, añade datos generados por ordenador sobre una imagen real [6]; Barcelona y Sevilla, además participan en las rutas turísticas guiadas mediante el dispositivo *Past View*, que permite al visitante adentrarse en el pasado de estas ciudades a través de reconstrucciones virtuales y tecnología de RA [7].

1.2 Justificación

Como ya se ha comentado en el presente documento, la Realidad Virtual es una tecnología en continuo crecimiento; una herramienta que, indudablemente, en un futuro, será una pieza elemental en nuestro día a día, siendo base y precursora de nuevas tecnologías más avanzadas que con el tiempo irán surgiendo. Es por ello que se cree de vital importancia que el Ayuntamiento de una ciudad de las características de San Cristóbal de La Laguna se vea inmersa y participe directamente en el desarrollo y crecimiento de aplicaciones y herramientas de este tipo, aprovechándose además de todas las ventajas que esta le puede brindar, como lo es el atractivo de esta tecnología que, sumada a la gran riqueza turística y cultural del municipio, podría suponer un aumento significativo de visitantes.

1.3 Objetivos

El objetivo general de este Trabajo de Fin de Grado es obtener una aplicación móvil de Realidad Virtual que contribuya a la difusión de la historia y el patrimonio cultural de la ciudad de San Cristóbal de La Laguna. Mediante esta, el usuario final podrá verse inmerso en un entorno 3D que simula la Plaza del Adelantado, pudiendo además moverse libremente por la misma. Para la consecución de este, se han planteado los siguientes objetivos específicos:

- Realizar una recopilación y estudio de información histórica del municipio, así como identificar a los actores y elementos que intervienen

en esta.

- Modelar con cierto nivel de detalle el entorno 3D que constituirá el escenario principal y los aledaños de este.
- Implementar las mecánicas de juego necesarias para hacer de la aplicación disfrutable para el usuario, además de accesible para el mayor número de usuarios posible, independientemente de sus características y/o disfuncionalidades.
- Optimizar la aplicación para su correcto funcionamiento en dispositivos móviles compatibles con *Samsung Gear VR*.

Capítulo 2
Herramientas y Tecnologías

2.1 Samsung Gear VR

Se ha utilizado un sistema HMD *Samsung Gear VR (2017) con Control Remoto* (*Gear VR* a partir de ahora) para el desarrollo y testeo de la aplicación, y como sistema objetivo sobre el que se ejecuta la misma.

Las *Gear VR* son un casco de Realidad Virtual desarrollado por *Samsung Electronics* en colaboración con *Oculus Rift*. La unidad funciona conectando vía MicroUSB o USB-C ciertos modelos de teléfono de *Samsung* de gama alta, como el *Galaxy S6*, *S7*, *S8*, y algunos modelos del *Galaxy Note*. Para el desarrollo de este proyecto, se ha utilizado un *Samsung Galaxy S7* con las características generales reflejadas en la Tabla 2.1.

Característica	Valor
Modelo	SM-G930F
Sistema Operativo	Android O (8.0.0)
Tamaño de pantalla	1440 x 2560 pixels, 5.1 pulgadas
Memoria	32GB ROM + 4GB RAM
CPU	2.3GHz Quad-Core (Custom Core) + 1.6GHz Quad-Core (Cortex®-A53)
GPU	ARM Mali-T880MP12 a 650MHz
Sensores	Acelerómetro, Giroscopio, Magnetómetro, otros.

Tabla 2.1: Características *Samsung Galaxy S7 utilizado*

Las *Gear VR* nos sumergen en una experiencia totalmente inmersiva que, junto al mando de control remoto (que se conecta al dispositivo móvil vía *Bluetooth*), nos da un mayor grado de interactividad, ya que este posee también los sensores necesarios para que la aplicación conozca su situación en cada momento (alabeo, cabeceo y guiñada), permitiéndonos así apuntar en cualquier ángulo del entorno virtual. Con esto se evita el tener que apuntar directamente con la mirada al elemento con el que se desea interactuar, resultando en una experiencia más realista para el usuario. En la Figura 2.1 se muestra un esquema de los diferentes botones del control remoto de las *Gear VR*.

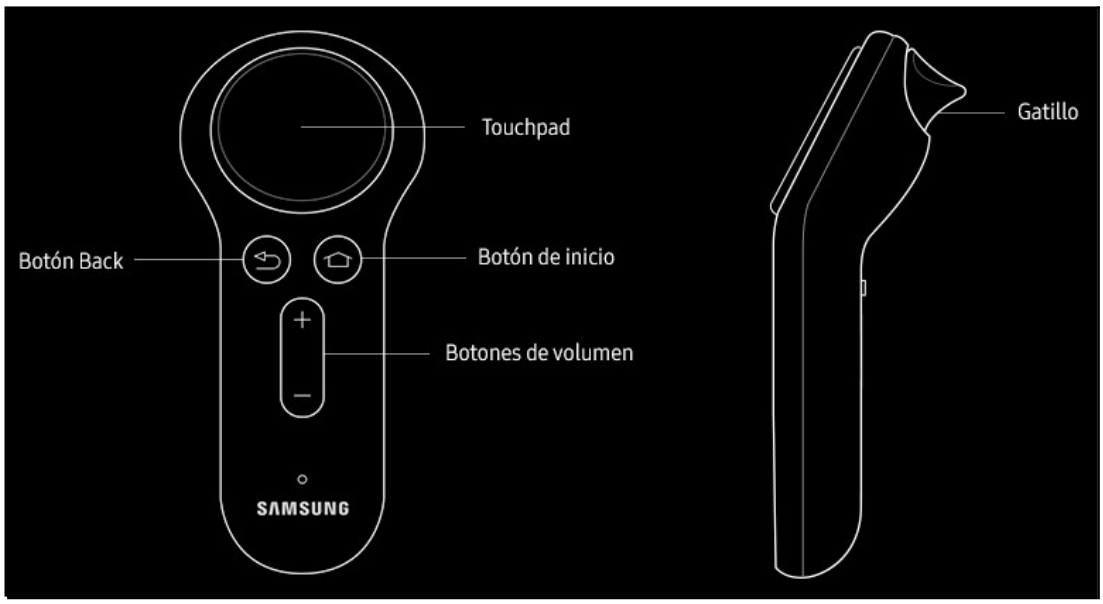


Figura 2.1: Botones del controlador remoto de Samsung Gear VR [8]

2.2 Unity 3D (v2017.4.0f1)

Unity 3D (a partir de ahora *Unity*), es un motor de videojuegos multiplataforma creado por *Unity Technologies*. *Unity* está disponible como plataforma de desarrollo para entornos *Microsoft Windows*, *OS X* y *Linux*. Esta tiene soporte de compilación con diferentes tipos de plataformas, tanto móviles como de escritorio, entre las que destacaremos las relativas a dispositivos de Realidad Virtual por ser esta la temática del presente documento. Estos dispositivos son: *Oculus Rift, HTC Vive*, *PlayStation VR*, *Microsoft Hololens*, *Google Cardboard*, y *Samsung Gear VR*, siendo estos dos últimos los más empleados en Realidad Virtual móvil.

Unity ha sido la plataforma escogida para el desarrollo de la aplicación por su facilidad de uso en el desarrollo, testeo y construcción de la misma, así como por ofrecer una versión de licencia gratuita (siempre que la compañía no perciba más de 10.000USD en ingresos brutos anuales o en recaudación de fondos), la cual proporciona (además de las prestaciones básicas del motor, actualizaciones continuas y acceso a la versión beta) herramientas de análisis básico, multijugador de hasta 20 jugadores simultáneamente, desarrollo para todas las plataformas y compras dentro de la aplicación entre otros. Esta licencia gratuita también nos garantiza que tenemos la propiedad absoluta del contenido que creemos (sujetos a licencias de paquetes externos que se utilicen en los proyectos). Además, en la red se encuentra una gran cantidad de recursos, ya sean tutoriales, paquetes de desarrollo o foros y comunidades enteras dedicadas al propio *Unity*.

2.2.1 C#

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por *Microsoft* como parte de su plataforma *.NET*. Su sintaxis básica deriva de *C/C++* y utiliza el modelo de objetos de la plataforma *.NET*, similar al de *Java*, aunque incluye mejoras derivadas de otros lenguajes.

Este lenguaje de programación se ha utilizado para programar las mecánicas de juego que no son tan sencillas de hacer desde la interfaz gráfica del editor de *Unity*, y para muchas otras tareas que, debido al amplio conocimiento de programación, pero no de diseño 3D, se ha creído más simple y cómodo realizarlas desde código.

Han sido diversos los motivos por los que se ha decidido utilizar *C#* antes que otros lenguajes como podría ser *JavaScript*. Los más determinantes han sido la futura eliminación del compilador de *JavaScript* del editor de *Unity*, la robustez, modernidad y seguridad de este lenguaje, y la similitud con los lenguajes *C/C++*, ampliamente trabajados en muchas de las asignaturas del Grado en Ingeniería Informática.

2.2.2 TextMesh Pro (1.2.2)

TextMesh Pro [9] es una herramienta para *Unity* que reemplaza componentes próximamente obsoletos como el *TextMesh* y el *UI Text*. *TextMesh Pro* utiliza una técnica de renderizado conocida como *Signed*

Distance Field, lo que hace que hace que el texto sea renderizado de forma limpia a cualquier tamaño y resolución.

Esta herramienta permite a los usuarios tener un mayor control sobre los caracteres, palabras, líneas, espacio entre párrafos y soporte para kerning, entre otros. Además, está optimizada tanto para PC como para plataformas móviles, lo cual es sumamente importante para el presente proyecto.

TextMesh Pro se ha utilizado principalmente para dar forma a los subtítulos y demás mensajes para el usuario, así como para el texto del menú de opciones.

2.2.3 ProBuilder (v2.9.8f3)

ProBuilder [10] es una extensión para *Unity* disponible desde la propia *Asset Store* que, desde febrero de este año 2018, forma parte de las herramientas oficiales de *Unity*, siendo ahora gratuita para todo tipo de usuarios.

Es una herramienta híbrida de modelado 3D y herramientas de diseño de niveles, optimizada para construir geometrías simples, pero también es capaz de ofrecer edición detallada y despliegue UV.

Se ha utilizado *ProBuilder* para el modelado de algunas de las edificaciones que rodean la Plaza del Adelantado, aprovechando la ocasión para probar la potencia de esta herramienta de prototipado rápido recientemente adquirida por *Unity*, que ha resultado cumplir con lo que promete, dando al diseñador una gran flexibilidad y facilidad de uso.

Como conclusiones sobre el uso de esta herramienta, destacar que puede ser una excelente opción para usuarios de *Unity* sin experiencia en el modelado 3D, por tener una interfaz gráfica muy intuitiva y fácil de usar, sin dejar de permitir la creación de modelos muy vistosos. Sin embargo, si se desea obtener modelos más elaborados, se recomienda utilizar otro software como podría ser *Blender*, del cual hablaremos más adelante en el presente documento.

2.2.4 Morph 3D MCS (v1.6)

Morph 3D [11] es una empresa que se dedica, sobre todo, a la innovación en espacios de Realidad Virtual. Esta empresa ha desarrollado un sistema de

Realidad Virtual, Aumentada y Mixta para la creación de avatares virtuales basados en la película de Steven Spielberg *Ready Player One* (basada en la novela homónima de Ernest Cline).

Además, *Morph 3D* ha desarrollado y publicado en la *Asset Store* de *Unity* diversos paquetes de extensión para la creación de personajes humanoides (la mayoría de pago), entre los que encontramos los paquetes *MCS Female* y *MCS Male* para la creación de personajes femeninos y masculinos respectivamente, ambos gratuitos. Los personajes creados con estas herramientas permiten una gran variedad de modificaciones de las características físicas, estando además optimizados para sistemas móviles y de Realidad Virtual.

Estas herramientas se han utilizado para la creación de los diferentes personajes que intervienen en la aplicación.

2.2.5 Standard Assets (v1.1.5)

Los *Standard Assets* de *Unity* son colecciones de assets, scripts y escenas de ejemplo, los cuales se pueden usar como base de un proyecto. Para este proyecto, se ha utilizado únicamente el paquete de personajes para la animación de los personajes de la aplicación.

2.2.6 VR Samples (v1.3)

Los *VR Samples* de *Unity* son, al igual que las *Standard Assets*, colecciones de assets, scripts y escenas de ejemplo, pero de Realidad Virtual. De esta herramienta se han utilizado principalmente assets y scripts como base para manejar las interacciones con los diferentes elementos del escenario de la aplicación.

2.2.7 Oculus Utilities (v1.26)

Las *Oculus Utilities* son un paquete personalizado desarrollado por el propio *Oculus*, que incluye útiles para *Unity*, o el plugin para *Unity* del *Platform SDK* de *Oculus*, entre otros.

De esta herramienta se han utilizado principalmente prefabs de modelos 3D, assets y scripts para la simulación del controlador remoto en la aplicación y como base para manejar mecánicas relacionadas con el personaje en primera persona y el seguimiento del controlador remoto.

2.3 QGIS (v2.18.18)

QGIS [12] es un Sistema de Información Geográfica (SIG) de código abierto licenciado bajo GNU – General Public License. Es un proyecto oficial de *Open Source Geospatial Foundation* (*OSGeo*), y soporta numerosos formatos y funcionalidades de datos de tipo vectorial, datos ráster y bases de datos.

Esta herramienta proporciona una amplia gama de capacidades a través de sus funciones básicas y complementos. Puede visualizar, gestionar, editar y analizar datos y diseñar mapas imprimibles.

QGIS ha sido empleado para generar un archivo ESRI Shapefile (SHP) del Casco Histórico de la ciudad de San Cristóbal de La Laguna con el que, gracias al software *Blender* (del que hablaremos más adelante), se ha generado un modelo 3D básico de esta ciudad.

2.3.1 LIDAR

El *LIDAR* [13] (de Light Detection and Ranging) es una técnica de teledetección óptica que utiliza la luz de un láser para obtener una muestra densa de la superficie de la tierra, produciendo mediciones exactas de x, y y z.

De esta manera, y para lo que nos interesa en este proyecto, se pueden obtener datos masivos y bastante exactos de las altitudes de diferentes puntos de un mapa entero (incluyendo alturas de edificios).

Se han utilizado datos *LIDAR* del Casco Histórico de San Cristóbal de La Laguna que, unidos a los datos catastrales (ambos obtenidos del Instituto Geográfico Nacional) del municipio, han servido para la generación de un archivo SHP con las alturas de las diferentes edificaciones de la ciudad para su posterior tratamiento en *Blender*.

2.4 COLMAP (v3.4)

COLMAP [14] es un pipeline de propósito general SfM (Structure from Motion) y MVS (Multi View Stereo) con interfaz gráfica y de línea de comandos. Ofrece una amplia gama de características para la reconstrucción de imágenes secuenciales y no secuenciales.

Con esta herramienta se han obtenido archivos OBJ con nubes de puntos de las diferentes edificaciones y demás elementos que conforman tanto la

Plaza del Adelantado como sus alrededores, con los cuales se pueden reconstruir dichas edificaciones y elementos.

2.5 Blender (v2.79a)

Blender [15] es un software dedicado especialmente al modelado, iluminación, renderizado, animación y creación de gráficos 3D. A finales del año 2002, esta herramienta pasó a una licencia de un software libre.

Se ha utilizado *Blender* para el modelado de algunas de las edificaciones que rodean la Plaza del Adelantado, siendo más complicada de usar que *ProBuilder*, pero generando resultados más elaborados y optimizados. Además, muchos de los modelos realizados con *ProBuilder* fueron importados en *Blender* para su optimización con esta herramienta.

Blender también se ha empleado para importar el archivo SHP generado con *QGIS* y los datos *LIDAR*, y obtener un modelo 3D del Casco Histórico haciendo uso del add-on *BlenderGIS*.

2.6 SQLite (v3.24.0)

SQLite [16] es un sistema gestor de bases de datos relacional contenida en una biblioteca relativamente pequeña, escrita en *C*. El motor de *SQLite* no es un proceso independiente con el que se comunica el programa, sino que la biblioteca *SQLite* se enlaza con este, pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de *SQLite* a través de llamadas simples a subrutinas y funciones, lo que reduce la latencia en el acceso a la base de datos, ya que las llamadas a funciones son más eficientes que las comunicaciones entre procesos. El conjunto de la base de datos es guardado como un único fichero estándar en la máquina host, diseño simple que se logra bloqueando todo el fichero al principio de cada transacción.

Este sistema gestor se ha utilizado en este proyecto para la implementación de subtítulos, almacenando los fragmentos de texto de estos, así como los tiempos que los mismos permanecen en pantalla.

Se ha optado por utilizar SQLite, por ser sumamente fácil de instalar e integrar con Unity, además de por ocupar muy poco espacio aun contando con toda la potencia de los comandos DDL y DML de SQL.

2.7 Python (v.3.6.5)

Python [17] es un lenguaje de programación interpretado, multiplataforma y con tipado dinámico cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Además, se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional.

Este lenguaje de programación ha sido utilizado en el proyecto para la automatización de la creación y carga de la base de datos en la que se alojan los subtítulos de la aplicación.

Se ha optado por emplear *Python* frente a otros lenguajes de programación por su gran facilidad de uso e integración con *SQLite* gracias a la librería "sqlite3".

Capítulo 3
Desarrollo

3.1 Formación inicial

La primera tarea a afrontar antes de comenzar el proyecto como tal, ha sido la formación inicial, la cual ocupó los primeros meses previos al desarrollo para adquirir los conocimientos básicos de las herramientas que se pretendía emplear. En este período se estudiaron las bases teóricas y prácticas de las herramientas y tecnologías a utilizar, principalmente de *Unity 3D* y de *Blender*, así como las buenas prácticas para el desarrollo de aplicaciones de Realidad Virtual.

Para ello se utilizaron los propios recursos y apuntes de la asignatura Interfaces Inteligentes para conocer las bases de *Unity* y algunas buenas prácticas para el desarrollo de aplicaciones de este tipo, así como también se utilizó la guía de buenas prácticas de *Google* y sendos tutoriales online que *Unity* pone a disposición de los usuarios.

3.2 Estudio histórico

Con el fin de ofrecer al usuario final una información de la mayor calidad posible, se ha realizado una recopilación de información histórica relacionada con el municipio.

Con la colaboración directa de D.ª Carmen Rosa Hernández Alberto (Gestora Cultural del Ayuntamiento de San Cristóbal de La Laguna), se ha redactado un documento con la información reseñable de las infraestructuras históricas de la Plaza del Adelantado, información que, finalmente, ha sido la incluida en la aplicación para aportarle la función de difusión histórica y cultural.

En este punto también se han recopilado documentos históricos digitalizados (como el primer plano que se conserva de la ciudad realizado por Leonardo Torriani, que se puede ver en la Figura 1.1) e imágenes representativas de la Plaza del Adelantado y las edificaciones históricas que la rodean.

3.3 Reconstrucción del Casco Histórico de la ciudad

La primera tarea de desarrollo en sí que se abordó, fue la reconstrucción del Casco Histórico de la ciudad como un modelo 3D. Para ello se han obtenido los datos *LIDAR* de San Cristóbal de La Laguna en formato tabla que, unidos a los datos catastrales de la ciudad, nos permiten obtener las alturas exactas de las diferentes parcelas de cada una de las edificaciones. Con estos datos y la ayuda del software *QGIS*, se ha generado un archivo SHP cuya representación, con una base cartográfica estándar de *Open Street Map* [18], se puede visualizar en la Figura 3.1.

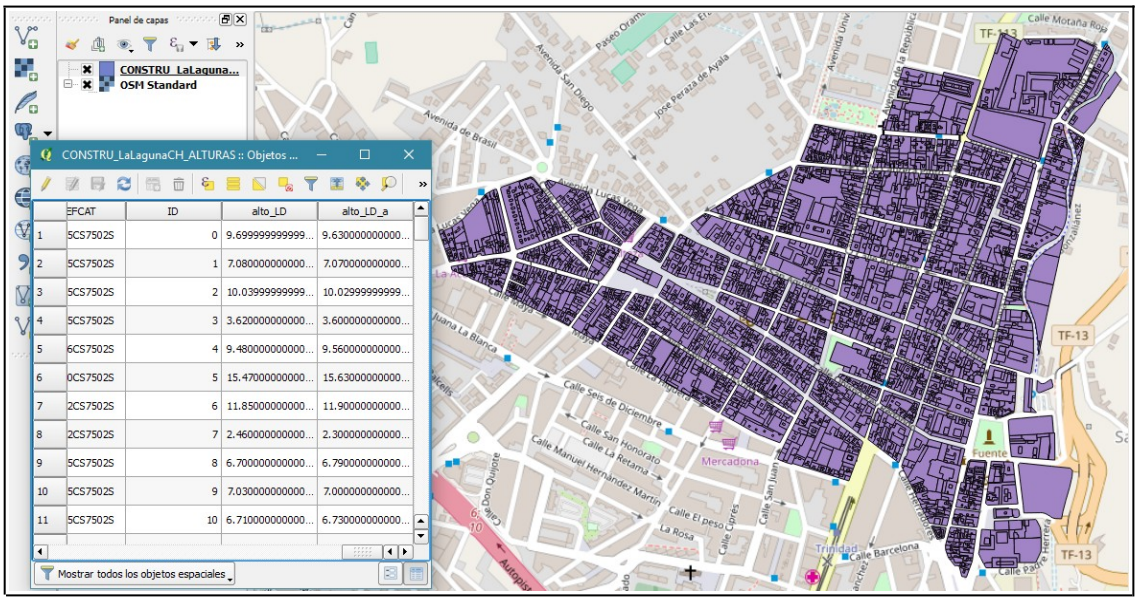


Figura 3.1: Representación del archivo SHP del Casco Histórico de la ciudad con base cartográfica OSM

Una vez obtenido este archivo SHP con las alturas, se ha importado en *Blender* gracias al add-on *BlenderGIS* [19], que permite la importación de archivos SHP y la auto extrusión de los polígonos por el valor del campo que se desee, así como el eje de la extrusión, tal y como se puede ver en la Figura 3.2, generando así un skyline realista.

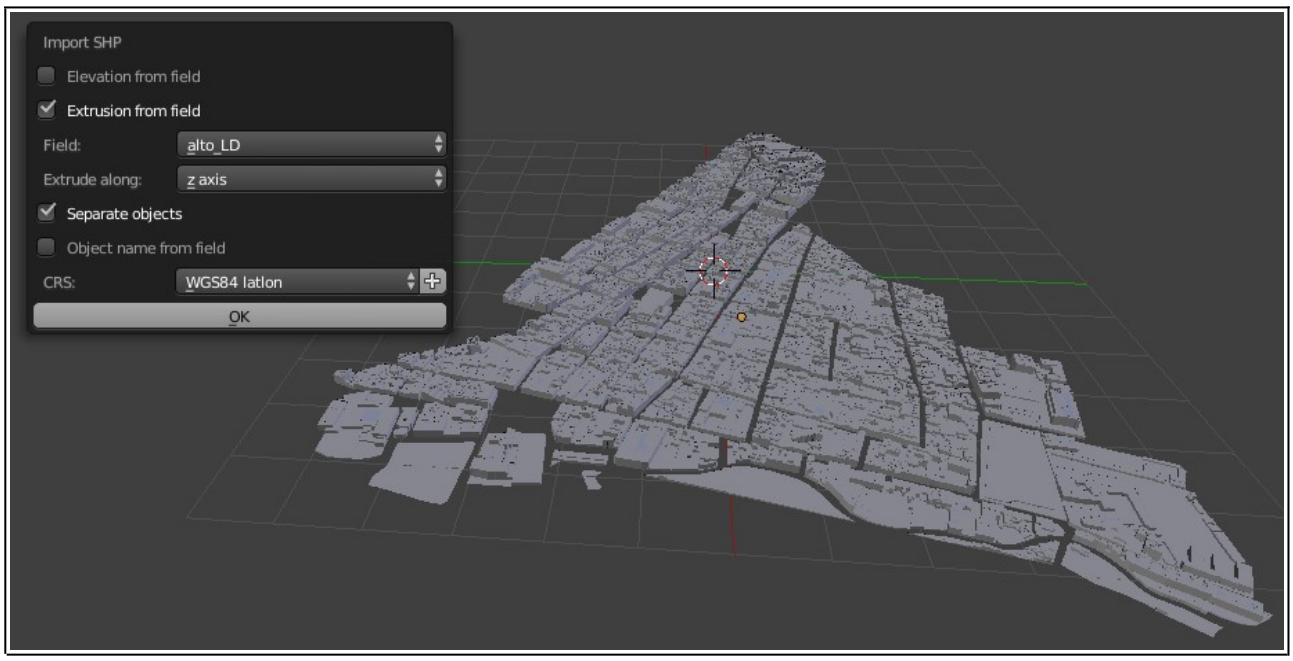


Figura 3.2: Importación a Blender del SHP de la ciudad con BlenderGIS

El modelo obtenido está compuesto por 5067 objetos (uno por cada registro del archivo SHP) separados porque se ha marcado la opción "Separate objects" a la hora de hacer la importación. Esto es interesante para poder eliminar y añadir diferentes objetos al modelo convenientemente, aunque para ello se deba sacrificar algo de rendimiento.

Este modelo se ha exportado con formato FBX (totalmente compatible con *Unity*) para que conserve la separación de los más de 5000 objetos, ya que si se hiciera, por ejemplo, con formato OBJ, se obtendría como un único objeto, lo cual no nos interesa. Una vez importado en *Unity* el FBX obtenido, se ha utilizado tanto para reconstruir la Plaza del Adelantado, como para realizar un modelo tipo maqueta con el que el usuario pueda apreciar, desde la aplicación, el trazado del plano en damero del Casco Histórico de la ciudad, tal y como se puede ver en la Figura 3.3.

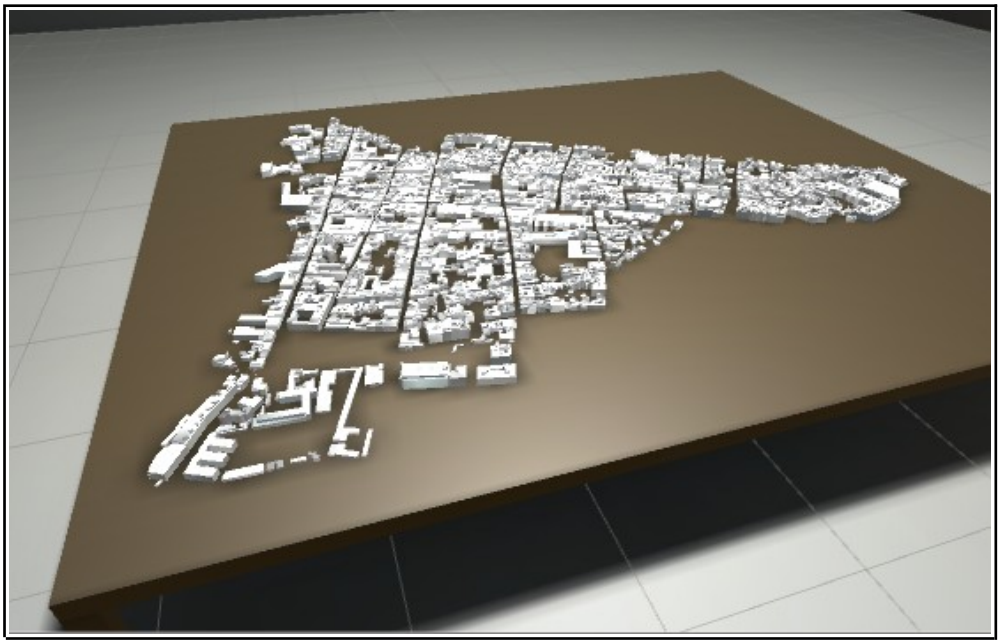


Figura 3.3: Modelo tipo maqueta del Casco Histórico de San Cristóbal de La Laguna

3.4 Modelado 3D

En este apartado se describen los procesos de modelado 3D que se han seguido en el proyecto para la creación de los diferentes elementos del escenario virtual en el que se desarrolla la aplicación.

3.4.1 Edificaciones

Como se puede apreciar en la Figura 3.2, al crear el modelo de la ciudad, las edificaciones son simples polígonos ortogonales que, más allá de su volumen, no se asemejan a las edificaciones reales a las que representan. Es por ello que se procedió a modelar con cierto nivel de detalle las edificaciones históricas que rodean la Plaza del Adelantado, así como unas viviendas inspiradas en la época para duplicar y repartir por todo el modelo de la ciudad.

En un primer momento se reconstruyeron con *Blender* algunas edificaciones con archivos OBJ de nubes de puntos de dichas edificaciones, obtenidos a través de la herramienta *COLMAP*, dando lugar a modelos de una sola malla y pendientes de optimizar que se debían separar manualmente en las diferentes partes de la edificación para, posteriormente, poder aplicar

materiales y texturas correctamente. Este resultó ser un trabajo aparentemente más tedioso que el propio modelado manual, por lo que se optó por este último método.

Para realizar el modelado, primero se usó la extensión *ProBuilder* de *Unity*, por lo que el modelado se realizaba directamente en el propio editor. No hay que olvidar que *ProBuilder* es una herramienta de prototipado, cuyos resultados no suelen estar optimizados ni del todo bien construidos (creando geometrías duplicadas, vértices que no terminan de unir, entre otros). A pesar de ser una herramienta de prototipado rápido, se optó finalmente por terminar de modelar las edificaciones con *Blender*, así como importar en este los modelos realizados con *ProBuilder* para optimizarlos correctamente.

El modelado en *Blender* se realizó de manera simple con, principalmente, extrusión y escalado de geometrías, intentando siempre respetar los principios del diseño con quads. Además, se utilizaron modificadores "subdivision surface" para el suavizado de las esquinas. Posteriormente, ya desde *Unity*, se aplicaron materiales y texturas a las diferentes mallas de las edificaciones para darles un aspecto más realista. A continuación se pueden ver algunas versiones de los modelos 3D realizados, unos con más detalles que otros según el software de modelado utilizado.

Figura 3.4: Primera versión del modelo 3D del Ayuntamiento de San Cristóbal de La Laguna (Blender)

Figura 3.5: Ermita de San Miguel (ProBuilder)

Figura 3.6: Fuente de la Plaza del Adelantado (Blender). De fondo el Convento de las Dominicas (ProBuilder) y el Palacio de Nava y Grimón (Blender)

Figura 3.7: Casa de Anchieta (Blender) y Hotel Nivaria (ProBuilder)

Figura 3.8: Calle Nava y Grimón, ejemplo perfecto de las calles de La Laguna que casi se pierden en el horizonte

3.4.2 Personajes

Dentro de la aplicación, el usuario podrá interactuar con cada uno de los personajes que se encuentran en el escenario. Cada personaje está asociado a una edificación diferente, de la que se dará información una vez se interactúe con el respectivo personaje. Estos se han colocado en la escena de forma que el usuario pueda interactuar con ellos sin dejar de visualizar la edificación

correspondiente.

Se han utilizado las herramientas de *Morph 3D MCS Female* y *MCS Male* para la generación automática de modelos humanoides femeninos y masculinos respectivamente, los cuales de pueden observar en la Figura 3.9. Cada uno de estos modelos se ha asignado como avatar dentro de un prefab "AIThirdPersonController" de los *Standard Assets* de *Unity*, de forma que se autodetecta el esqueleto del modelo de *Morph 3D* y se anima con las mismas animaciones de los *Standard Assets* que hacen lo propio con *Ethan* (el avatar por defecto de los *Standard Assets* de *Unity*).

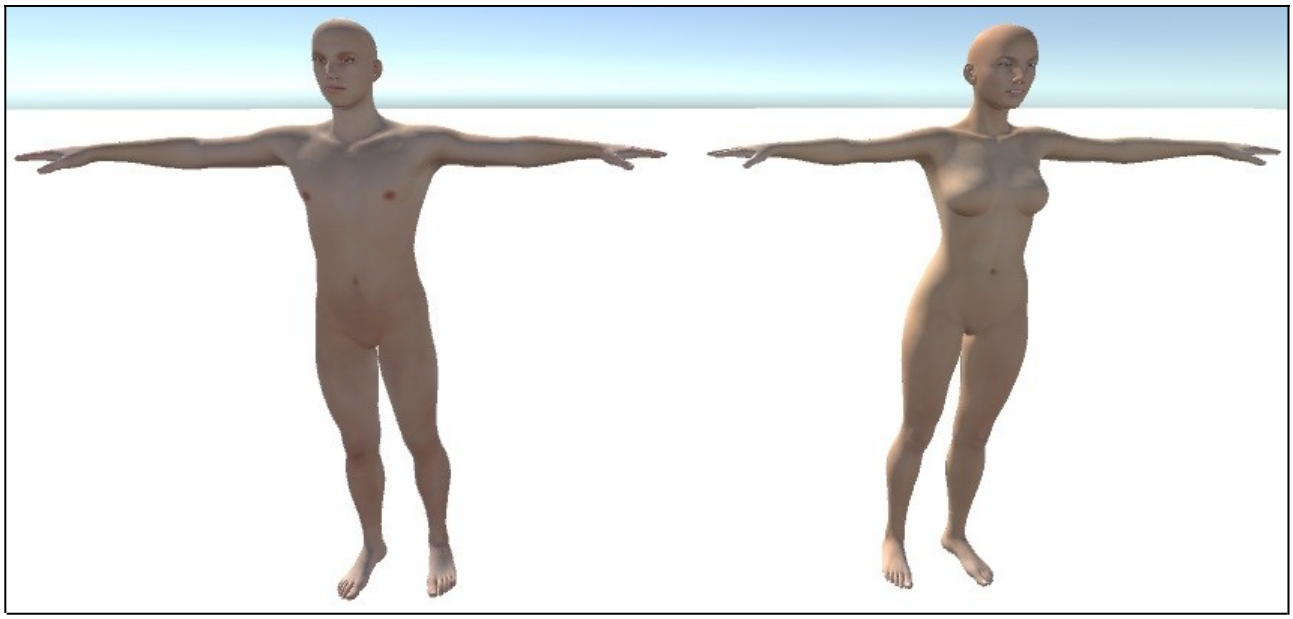


Figura 3.9: Modelos base de Morph3D MCS Male y MCS Female de izquierda a derecha

Estos avatares serán simplemente personajes por defecto para que el usuario pueda realizar las interacciones, pero deberán ser cambiados en un futuro por avatares acordes a la época, sencillamente modelando el personaje y haciendo dicho modelo hijo del "AIThirdPersonController" correspondiente, de forma que tendrá automáticamente las animaciones de este.

3.5 Realidad Virtual

El primer paso para trabajar con un proyecto de Realidad Virtual para *Android* en *Unity* es cambiar la plataforma objetivo de compilación a *Android* desde la Configuración de Compilación. Además, desde la Configuración del Jugador se debe añadir el SDK de Realidad Virtual de *Oculus* (también se ha añadido la opción "None" para las ejecuciones en dispositivos que no soporten

el SDK de *Oculus*). Por último, se debe añadir un "Oculus Signature File" para el desarrollo para *Gear VR*. Para ello se necesita el identificador del dispositivo móvil que se va a utilizar para testeo, generar el archivo desde el propio sitio web de *Oculus* y guardarlo en en directorio *Assets/Plugins/Android/assets* del proyecto (si se necesita testear en más de un dispositivo, se debe incluir el fichero generado para cada uno de ellos o la aplicación no funcionará). Con esto, ya se puede compilar el proyecto para el dispositivo móvil en cuestión y utilizarlo para testeo. Destacar que, si se va a generar la versión de lanzamiento para su publicación en *Oculus Store*, se debe comprobar que previamente se han retirado todos los ficheros de firma osig del proyecto, ya que las aplicaciones descargadas a través de *Oculus Store* se pueden ejecutar en cualquier dispositivo *Gear VR*.

En este apartado se describen los diferentes elementos que constituyen el componente de Realidad Virtual de la aplicación y la forma en que se ha implementado cada uno de ellos.

3.5.1 Cámara

En este tipo de aplicaciones, la cámara se corresponde con el personaje en primera persona que controla el jugador. *Unity*, cuando crea una aplicación de Realidad Virtual, automáticamente implementa tanto el seguimiento de la cabeza como la doble imagen para la visión estereoscópica, quitando estas responsabilidades al desarrollador.

Los *VR Samples* de *Unity* contienen un prefab "MainCamera" constituido por una cámara con un raycast y un canvas para renderizar tanto la retícula como la barra circular de selección. Con estos elementos se permite la selección de objetos interactuables mediante el punto al que mira el usuario, gracias a los diferentes scripts asociados al objeto padre del prefab. Este se ha utilizado como cámara principal del jugador, aprovechando además las funcionalidades para la interacción con los elementos de la escena.

3.5.2 Controlador

En la "MainCamera" de los *VR Samples* se tiene una cámara con los elementos necesarios para la interacción con los diferentes elementos, pero esta cámara es estática, mientras que se desea que el jugador se pueda desplazar libremente por el entorno.

Para ello, las *Oculus Utilities* contienen un prefab "OVRPlayerController" constituido por una cámara y diversos scripts de control de movimiento, entre otros. Este se ha utilizado como controlador del personaje en primera persona, sustituyendo su objeto hijo "CenterEyeAnchor" (el cual contiene la cámara del controlador) por el prefab "MainCamera" de los *VR Samples*. De esta forma se tiene un controlador en primera persona con movimiento ("OVRPlayerController") y las funcionalidades necesarias para interactuar con el entorno ("MainCamera"). Destacar que el prefab "MainCamera", una vez añadido a la jerarquía del controlador, se ha renombrado a "OVRPlayerController" para mantener los nombres del prefab de Oculus.

Además, este controlador permite añadir modelos 3D que simulen las manos o, en este caso, mandos controladores, de forma que en la aplicación se ve el controlador remoto de la misma forma que se vería en la realidad. Las *Oculus Utilities* contienen un prefab "TrackedRemote" que contiene un modelo 3D tanto del controlador de *Oculus Go* como del controlador de *Gear VR*. Basta con hacer este prefab hijo tanto de "LeftHandAnchor" como de "RightHandAnchor" y ajustar el valor "Controller" en el inspector según se trate del controlador para la mano derecha o izquierda. Esto se hace dentro del "OVRPlayerController", el cual se encarga de detectar si el jugador está usando un controlador de *Oculus Go* o de *Gear VR* y si lo tiene configurado como mano izquierda o derecha y mostrarlo convenientemente.

Por último, para configurar el movimiento del personaje en primera persona con el controlador remoto, se ha modificado la función *UpdateMovement()* del "OVRPlayerController", añadiendo la detección del panel táctil en sus dos ejes (x e y).

3.5.3 Raycast

El raycast es el láser que sale desde el centro de la mirada del personaje en primera persona, con el cual se interactúa con el entorno. Como se ha mencionado anteriormente en el presente documento, se ha querido que el raycast no dependa de la mirada del jugador, sino que pueda ser manejado con el controlador remoto que se ha configurado en el punto anterior.

Para ello, se ha modificado el script "VREyeRaycaster" del nuevo "CenterEyeAnchor" del controlador del jugador, primero añadiendo:

- Una referencia privada a un componente "LineRenderer" usada para

visualizar el raycast (visible desde el inspector).

- Un booleano público que controlará si el raycast es visible o no.
- Una referencia privada al "TrackingSpace" del equipo de cámara, a la que será relativo el raycast.
- Un getter que consulta si hay un controlador remoto conectado o no.
- Un getter que devuelve el controlador remoto conectado (si lo hubiese).

Una vez hecho esto se ha añadido a la función *EyeRaycast()* dos vectores tridimensionales (*Vector3*) que representan el inicio y el fin del raycast, y este se activa o desactiva si hay un controlador remoto conectado o no. También se crea el nuevo láser, relativo a la posición del controlador remoto, si lo hubiese, y se controla si el nuevo láser colisiona con algún objeto interactuable, en cuyo caso se establece el final del láser al punto en el que colisiona con dicho objeto. Además, se modifica la posición de la retícula ("GUIReticle") para que coincida con el final del raycast. El resultado se puede apreciar en la Figura 3.10.

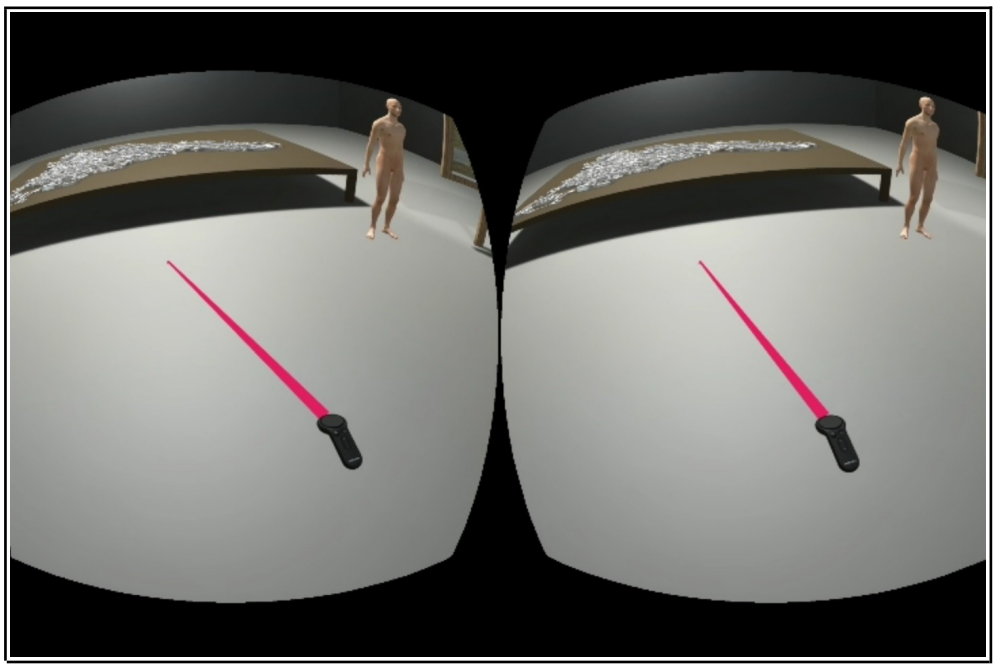


Figura 3.10: Resultado raycast

Para que todo esto funcione, se han asociado a los diferentes atributos de los scripts modificados los objetos y assets correspondientes desde el inspector

de *Unity*, como se observa en la Figura 3.11.

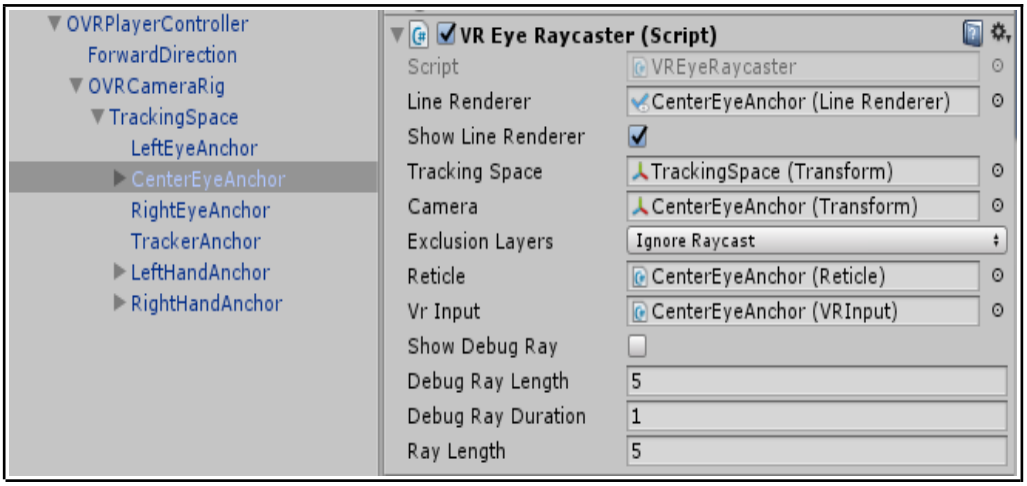


Figura 3.11: Asociación de objetos en inspector de Unity para el raycast

3.6 Implementación de acciones y mecánicas

En este apartado se describen las acciones y mecánicas que se han implementado en el proyecto para la correcta interacción del usuario con los diferentes elementos del escenario virtual en el que se desarrolla la aplicación.

3.6.1 Interacción con personajes

Los elementos principales con los que podrá interactuar directamente el usuario para activar o desactivar determinadas acciones en la aplicación, son los personajes (avatares humanoides). Para poder interactuar con estos mediante el raycast, los *VR Samples* nos ofrecen un script "VRInteractiveItem", el cual, asociándolo a cualquier objeto que además posea un collider, hace a dicho objeto interactuable en el volumen o superficie que cubra su respectivo collider. De esta forma, asignando este script al objeto más alto en la herencia del personaje (al cual se le ha configurado el collider como se describe en el apartado *3.6.2 Colliders*), se consigue que el personaje sea un objeto interactuable mediante el raycast.

Una vez que el personaje es interactuable, se deben establecer las acciones mediante las cuales se va a interactuar con él. Para ello, el objeto interactuable se encuentra suscrito a una serie de eventos que interpretan las diferentes acciones del raycast con dicho objeto. Los eventos utilizados en el proyecto han sido los descritos a continuación:

- *OnOver*: Se activa cuando el raycast apunta directamente al collider del objeto en cuestión. Activa la posibilidad de interacción con el objeto ya que, de lo contrario, la posibilidad de interacción estará siempre desactivada. Además, cambia la imagen de la retícula y reproduce un sonido representativo, lo que facilita al usuario la identificación del objeto interactuable.

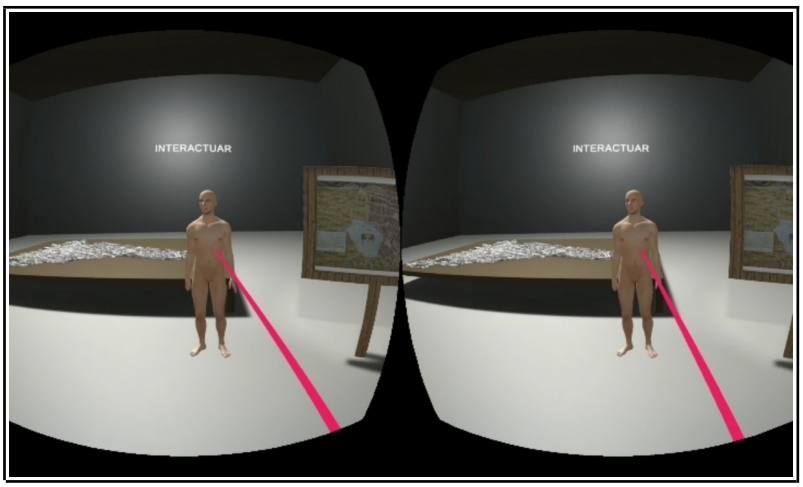


Figura 3.12: Ejemplo del evento OnOver

- *OnOut*: Se activa cuando el raycast deja de apuntar directamente al collider del objeto en cuestión. Desactiva la posibilidad de interacción con el objeto y cambia la imagen de la retícula por la original.

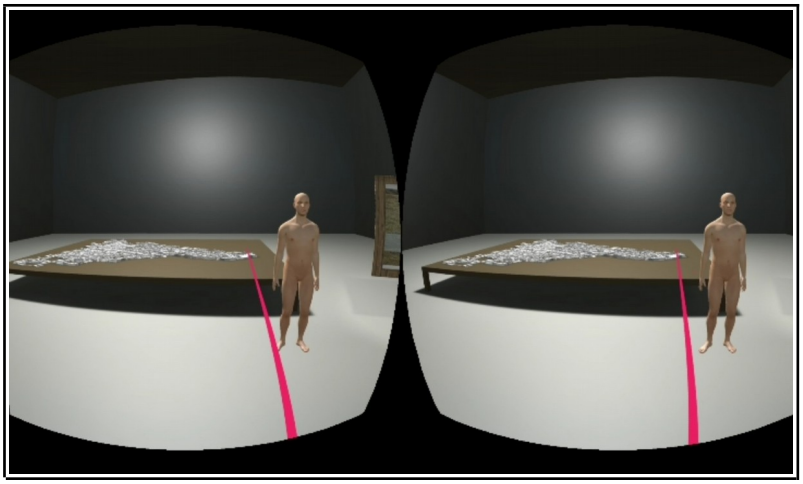


Figura 3.13: Ejemplo del evento OnOut

- *OnSelectionComplete*: Se activa cuando se mantiene pulsado el botón de interacción (el gatillo en este caso) y la posibilidad de interacción está activada. Este evento activa la interacción con el objeto, reproduciendo

la información de la edificación a la que está asociada. Además, bloquea el desplazamiento del usuario por la escena y activa el panel de imágenes (el cual se describe en el apartado *3.6.3 Paneles de imágenes*). Este evento también se utiliza para desactivar la interacción con el objeto una vez iniciada.

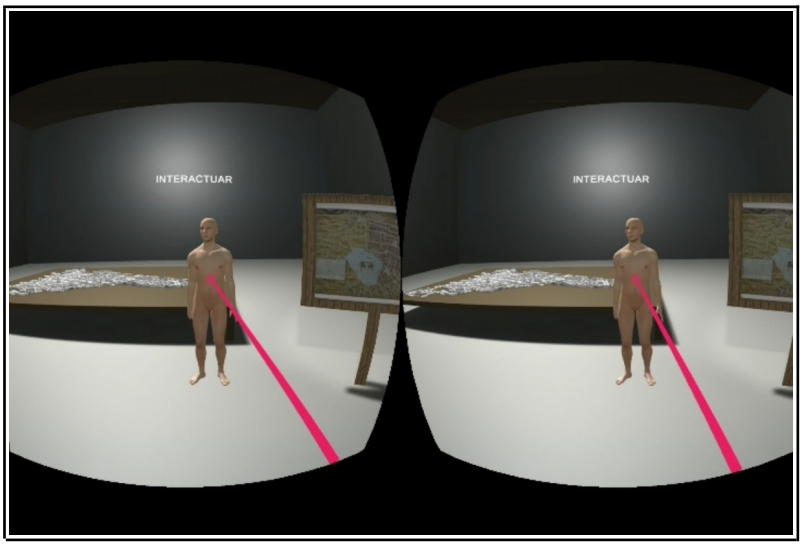


Figura 3.14: Ejemplo de progreso del evento OnSelectionComplete

3.6.2 Colliders

Se han utilizado colliders para simular las físicas de colisión tanto con las edificaciones como con los personajes y demás elementos del escenario. Estas físicas no pretenden simular fuerzas como podrían ser la gravedad o el empuje, sino las colisiones del usuario con objetos rígidos, ya que los objetos de la escena son estáticos y no precisan de estas fuerzas. En la medida de lo posible, se ha intentado evitar el uso de mesh colliders con el objetivo de no sobrecargar los cálculos en caso mallas complejas. También, para los personajes, se han utilizado capsule colliders, usualmente utilizados en videojuegos como colliders de personajes.

Además, se ha incluido un collider que cubre toda la Plaza del Adelantado, delimitando así el perímetro por el que se permite el desplazamiento del usuario, es decir, el jugador no podrá salir del interior del collider.

3.6.2.1 Colliders como disparadores

Es de gran interés que cuando el usuario interactúe con algún personaje y

reciba información histórica de algún elemento, el ángulo de visión sea el adecuado para poder ver completamente tanto el elemento del cual se está recibiendo información, como el panel de imágenes asociado al mismo.

Para ello, estos han sido colocados en la escena en puntos desde los que, si el usuario interactúa con el personaje situándose frente al mismo, el ángulo sea el adecuado. Como no se puede esperar que el usuario siempre haga la interacción en el punto adecuado, se han situado colliders frente a los personajes que actúan como disparadores, activando o desactivando el script "PlayInteraction" que controla las interacciones con los diferentes personajes, de forma que el jugador solamente pueda interactuar con estos desde un ángulo adecuado para la correcta visualización de los elementos que nos interesan, no pudiendo interactuar, por ejemplo, desde detrás del personaje ni desde un costado.

3.6.3 Paneles de imágenes

Se ha querido mostrar al usuario imágenes reales de las edificaciones y elementos históricos de la Plaza del Adelantado, ya sean del interior de la edificación en cuestión, o de las diferentes apariencias de la edificación en la historia, entre otros.

Para ello, se ha modelado un panel estándar que se ha replicado para cada una de las edificaciones y elementos históricos del escenario. Estos paneles se sitúan junto a cada uno de los personajes interactuables, de forma que, una vez iniciada la interacción, sean visibles además de la edificación o elemento en cuestión.

Estos paneles irán mostrando imágenes que acompañan y apoyan la información histórica que se esté recibiendo en cada caso. En la Figura 3.15 se puede observar un panel de imágenes en acción.

Figura 3.15: Panel de imágenes

3.7 Accesibilidad

El prototipo de aplicación resultado del presente proyecto pretende ser accesible para el mayor número de personas posible, por lo que se ha decidido maximizar la accesibilidad para todas aquellas personas con alguna disfuncionalidad abordable en una aplicación de estas características. En este caso concreto, se puede hacer accesible la aplicación a personas con alteración de las capacidades auditivas.

3.7.1 Menú de opciones

Se ha desarrollado un menú de opciones simple, de botones de interacción mediante el raycast tratado en el apartado *3.5.3*, mediante el cual se permite al usuario activar y desactivar algunas opciones de la aplicación, las cuales se describen a continuación:

- *Música*: Permite activar o desactivar el sonido ambiente de la aplicación.
- *Voces*: Permite activar o desactivar las explicaciones por voz que se dan al usuario. Si se intenta desactivar cuando la opción *Subtítulos* esté también desactivada, esta última se activará automáticamente.
- Efectos: Permite activar o desactivar los efectos de sonido relacionados

con las interacciones, como activar una opción del menú, o apuntar con el raycast a un objeto interactuable.

- *Mensajes*: Permite activar o desactivar los mensajes explicativos que se dan al usuario. Se recomienda que estén siempre activados.
- *Subtítulos*: Permite activar o desactivar los subtítulos de las explicaciones o información que se da al usuario. Si se intenta desactivar cuando la opción *Voces* esté también desactivada, esta última se activará automáticamente.

Todas estas opciones se pueden ver en la Figura 3.16, en la que se muestra el menú que se ha desarrollado.

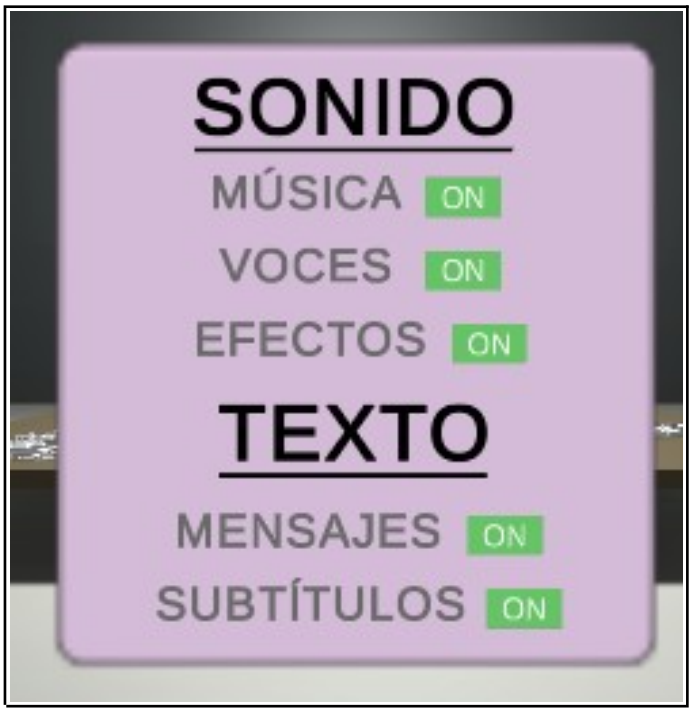


Figura 3.16: Menú de opciones

3.7.2 Subtítulos

Como se ha visto en el apartado anterior, mediante el menú de opciones se pueden activar o desactivar los subtítulos. Estos se muestran para las explicaciones o información que se dan al usuario, dando así una alternativa a la recepción por voz de esta información.

Se ha utilizado una base de datos *SQLite* local a la aplicación para almacenar los subtítulos y los tiempos de permanencia en pantalla de cada

uno de los fragmentos de estos. Para la automatización de la creación y carga de la base de datos, se ha implementado un pequeño script en *Python* que recibe como entrada un número entero que será la longitud máxima de los subtítulos en pantalla (en este caso, se ha establecido a 50), y un fichero de texto con todo el texto de los subtítulos así como los tiempos de permanencia en pantalla. El script crea una nueva tabla en la base de datos, parsea todo el texto y lo divide en fragmentos de longitud máxima especificada, y los inserta en la base de datos.

La recuperación de los subtítulos de la base de datos se realiza en tiempo de ejecución, dependiendo del objeto con el que se haya interactuado o de la información que se vaya a dar al usuario.

3.8 Optimización y gestión de los recursos

Se debe tener muy en cuenta los recursos que tendrán los dispositivos para los cuales se está desarrollando. En este caso se ha trabajado con un smartphone que, a pesar de ser de gama alta y ofrecer unas relativamente altas prestaciones, no deja de ser un dispositivo móvil, cuyos recursos distan mucho de los que podría tener un sistema de realidad de sobremesa como un PC o una videoconsola.

Por ello, se ha tratado de optimizar la gestión y el uso de los recursos hardware necesarios para la ejecución fluida de la aplicación desarrollada.

3.8.1 Audios

Se han utilizado audios tanto para las descripciones que el usuario recibe de los elementos históricos como para los efectos de sonido de interacciones con el entorno y el menú. Estos audios han sido obtenidos originalmente en diferentes formatos (.mp3, .m4a) que ocuparían más espacio del deseado en el dispositivo objetivo.

Por ello, se ha decidido convertir los audios generados y descargados a formato .ogg, ampliamente utilizado en videojuegos por su excelente relación calidad-tamaño. Con esto se ha conseguido una reducción considerable del tamaño final tanto del instalable como de la memoria ocupada en el smartphone una vez instalada.

3.8.2 Texturas y materiales

Se ha intentado utilizar el menor número de texturas y materiales repitiéndolos en los diferentes objetos de la escena en la medida de lo posible, tratando de no generar una escena aburrida y monótona. El objetivo de esta medida ha sido evitar la creación de texturas y materiales innecesarios, ayudando también a la mejor fluidez de la ejecución al no tener que renderizar tantos materiales diferentes.

3.8.3 Static Batching

El *static batching* combina objetos estáticos en la escena en grandes mallas y los renderiza más rápidamente. Esto permite al motor de *Unity* reducir las llamadas *draw calls* para geometrías de cualquier tamaño mientras que compartan el mismo material (relacionado con lo visto en el apartado 3.8.3 Static Batching) y no se muevan en la escena.

Para ello, se han marcado como estáticos todos los objetos de la escena que no se mueven durante la ejecución. Este método aumenta ligeramente la memoria utilizada por la aplicación, sacrificio necesario para una mayor fluidez.

3.8.4 Level Of Detail (LOD)

Unity nos permite establecer diferentes niveles de detalle en el renderizado de un objeto dependiendo de la cercanía de este respecto a la cámara activa. Esto se consigue reduciendo el número de triángulos renderizados por objeto según aumenta la distancia entre el objeto y la cámara, lo que se traduce en una menor carga para el hardware del dispositivo, mientras que la pérdida de detalles en objetos lejanos a penas será perceptible.

Para ello se ha añadido el componente “LOD Group” a las casas presentes en la escena, ya que son estas las que representan la mayor carga gráfica de la aplicación, siendo además los únicos objetos que se pueden visualizar a lo lejos en las largas calles transversales de la Plaza del Adelantado.

3.8.5 Occlusiong Culling

Occlusiong Culling es una funcionalidad de *Unity* que desactiva el renderizado de los objetos que en cada momento no estén dentro del alcance

de la cámara (es decir, que no estén visibles), ya sea porque están detrás de la cámara o porque están ocultos detrás de otro objeto (funcionalidad que no se obtiene con *Frustum Culling*). Este proceso se hace en tiempo de ejecución, lo cual reduce drásticamente el número de objetos renderizados en la escena, reduciendo también la carga del hardware del dispositivo.

Para aprovechar esta funcionalidad, se han marcado como estáticos todos aquellos objetos en la escena que se quiere que no se rendericen si no están dentro del alcance de la cámara activa. Además se han creado *occlusing areas* en áreas en las que en algún momento podría haber objetos en movimiento. De esta forma se establece también esta funcionalidad para objetos no estáticos siempre y cuando se encuentren dentro del *occlusing area* definido.

3.8.6 Baked Lighting

En cualquier tipo de juego, las sombras siempre son algo fundamental, ya que sin ellas se pierde realismo, pero muchas veces costoso por los cálculos en tiempo de ejecución que estas requieren. *Unity* nos da la opción de generar previamente las sombras de objetos estáticos para que estas no tengan que calcularse en tiempo de ejecución, ya que calcular en cada fotograma las sombras de objetos estáticos es algo inútil.

Para esto, ya se tenía de apartados anteriores marcados como estáticos los objetos que no se mueven en la escena. Además se ha marcado la luz direccional (que emula la luz del sol) como mixta, de forma que permite calcular sombras de objetos no estáticos de en tiempo de ejecución, pero también permite la generación previa de las sombras de objetos estáticos.

Además, como las sombras de objetos estáticos no se van a calcular en tiempo real, se ha establecido la calidad de estas como alta. El tiempo que *Unity* tarda en generar estas sombras podrá tardar unas horas dependiendo del tamaño de la escena y de las características del equipo que estemos utilizando.

Capítulo 4
Conclusiones y líneas futuras

4.1 Conclusiones

Llegados a la parte final de este proyecto, pasaremos a recapitular y resumir los objetivos cumplidos con éxito, obteniendo así las correspondientes conclusiones.

Con la inestimable ayuda del Excmo. Ayuntamiento de San Cristóbal de La Laguna y, en especial, de D.ª Carmen Rosa Hernández Alberto (Gestora Cultural del Ayuntamiento de San Cristóbal de La Laguna), se ha obtenido una gran cantidad de información histórica de la ciudad, la cual ha quedado plasmada en la aplicación desarrollada.

Se ha diseñado y modelado un entorno 3D, con cierto nivel de detalle en el que el usuario tiene absoluto control de desplazamiento, pudiendo además recibir de forma visual gran parte de la información histórica que se ha recolectado del municipio.

Se han implementado diversas mecánicas para la total inmersión del usuario en el entorno 3D, teniendo las herramientas necesarias para interactuar con este de forma eficaz. Además, se ha conseguido preparar la aplicación para su uso por parte de personas con alguna disfuncionalidad auditiva gracias a un sencillo menú de opciones, sin que afecte negativamente a su experiencia de uso.

Se ha optimizado la aplicación para su eficiente funcionamiento en dispositivos móviles compatibles con *Samsung Gear VR*, donde la aplicación se ejecuta de forma fluida y sin interrumpir la experiencia de inmersión del usuario.

Una aplicación de estas características es una nueva forma para San

Cristóbal de La Laguna de difusión del patrimonio histórico y cultural del municipio, pudiendo además ser utilizada por el Ayuntamiento como un reclamo para un nuevo público de turismo más moderno y actual.

(El proyecto desarrollado se puede consultar y descargar en el siguiente enlace: https://github.com/alu0100821390/TFG_Kevin_Estevez_Exposito).

4.2 Líneas de trabajo futuro

Una posible línea de trabajo futuro es la ampliación del escenario 3D por el que se puede desplazar el usuario, pudiendo así mostrar más puntos históricamente relevantes de la ciudad y dar información sobre estos.

Otra posible línea de trabajo sería dar una mayor experiencia de juego al usuario, dando como resultado una aplicación más divertida y entretenida para usuarios de todas las edades. Por ejemplo, se podría implementar un sistema de puntos sobre la actual base de datos SQLite, basado en la recogida de residuos presentes por todo el escenario virtual, empleando e inculcando así unos valores que siempre ha defendido el municipio bajo el lema "La Laguna limpia".

También se podría y debería trabajar en un modelado 3D más profesional tanto de todo el escenario virtual como de los propios personajes incluidos en este para darle un aspecto más realista y adecuado respecto a la época histórica sobre la que se desarrolla la aplicación. Esto podría ser realizado en conjunto con un/a diseñador/a 3D experimentado/a, como lo podría ser un/a alumno/a del Grado en Bellas Artes de la Universidad de La Laguna. También sería conveniente sustituir las grabaciones actuales por algunas realizadas por dobladores profesionales.

Como última propuesta de línea de trabajo futuro, se propone el desarrollo de versiones para otros dispositivos y plataformas de Realidad Virtual, lo que supondría un mayor alcance sobre el público objetivo de la aplicación.

Capítulo 5
Summary and Conclusions

Once we have reached the final part of this project, we will recapitulate and summarize the successful objectives, obtaining the corresponding conclusions.

With the invaluable help of the City Hall of San Cristóbal de La Laguna and especially the help of Mrs. Carmen Rosa Hernández Alberto (Cultural Manager of the City Hall of San Cristóbal de La Laguna), a large amount of historical information about the city has been obtained, which has been reflected in the developed application.

A 3D environment has been designed and modeled with a certain level of detail in which the user has absolute control of displacement, and can also visually receive much of the historical information of the city that has been collected.

Various mechanics have been implemented for the total immersion of the user in the 3D environment, having the necessary tools to interact with the environment effectively. In addition, it has been possible to prepare the application to be used by people with hearing disability thanks to a simple options menu, without this disability adversely affect their use experience.

The application has been optimized for efficient operation on mobile devices compatible with *Samsung Gear VR*, where the application runs smoothly and without interrupting the user's immerion experience.

An application with these characteristics is a new way for San Cristóbal de La Laguna to disseminate the historical and cultural heritage of the municipality. It can also be used by the City Hall as a claim for a new and more modern tourist public.

(The developed project can be viewed and downloaded at the following link: https://github.com/alu0100821390/TFG_Kevin_Estevez_Exposito).

Capítulo 6
Presupuesto

La estimación del presupuesto para este proyecto se ha calculado, en parte, en función de una estimación del salario que podría percibir un ingeniero informático junior.

Tipo	Cantidad	Coste unidad	Coste total
Samsung Galaxy S7 (SM-G930F)	1ud.	549,00€	549,00€
Samsung Gear VR (2017) con Control Remoto	1ud.	129,00€	129,00€
Software	--	--	0,00€
Análisis e investigación	30h.	17,00€	510,00€
Diseño	50h.	12,00€	600,00€
Desarrollo	170h.	15,00€	2550,00€
Documentación	20h.	10,00€	200,00€
TOTAL			**4538,00€**

Tabla 6.1: Resumen de presupuesto

6.1 Justificación del presupuesto

El cálculo del presupuesto responde a los costes de los dispositivo y tecnologías hardware y software empleados, así como a las horas de trabajo empleadas en cada una de las tareas generales:

- Se ha utilizado un smartphone de gama alta y una gafas de Realidad Virtual, ambos de la marca Samsung. El coste de estas herramientas se

ha obtenido del sitio web del propio proveedor.

- Dado que en todo momento se ha utilizado software libre y software gratuito para desarrollar la aplicación, este apartado no constituye un gasto real para el proyecto.
- El gasto derivado de la mano de obra (es decir, de las horas de trabajo empleadas) se ha calculado en base a una estimación de lo que podría cobrar un ingeniero informático como desarrollador junior empleando su propio PC como herramienta principal de trabajo.
- La suma de todos los costes asociados al proyecto asciende a un total de 4538,00€, tal y como se puede observar en la Tabla 6.1.

Capítulo 7
Bibliografía

[1] **Memoria Digital de Canarias** – Fuente:

https://mdc.ulpgc.es/

[2] **Gobierno de Aragón – Departamento de Innovación, Investigación y Universidad.** (2015). *Análisis de tendencias: Realidad Aumentada y Realidad Virtual.* [PDF file] - Recuperado de:

http://www.aragon.es/estaticos/GobiernoAragon/Departamentos/InvestigacionInnovacionUniversidad/Areas/Sociedad_Informacion/Documentos/Estudio%20Prospectiva%20Analisis%20de%20tendencias%20RA%20y%20RV%20con%20formato.pdf

[3] **APP Turismo La Laguna** – Google Play:

https://play.google.com/store/apps/details?id=com.crokis.turismo

[4] **Barcelona in Gothic** – YouTube:

https://www.youtube.com/watch?v=ZUKTQu1f94I

[5] **Turismo de Salamanca** – Fuente:

https://www.salamanca.es/es/realidad-aumentada

[6] **Turismo de Galicia** – Fuente:

http://www.turismo.gal/inicio

[7] **PASTVIEW** – Fuente:

http://www.pastviewexperience.com/

[8] **Gear VR con Controller** – fnac:

https://www.fnac.es/espaciosamsung/gear-vr

[9] **TextMesh Pro** – Unity Asset Store:

https://assetstore.unity.com/packages/essentials/beta-projects/textmesh-pro-84126

[10] **ProBuilder** – Unity Asset Store:

https://assetstore.unity.com/packages/tools/modeling/probuilder-111418

[11] **Morph3D** – Fuente:

https://www.morph3d.com/

[12] **QGIS** – Fuente:

https://www.qgis.org/es/site/

[13] **ArcGIS.** (2016). ¿Qué son los datos LIDAR? - Recuperado de:

http://desktop.arcgis.com/es/arcmap/10.3/manage-data/las-dataset/what-is-lidar-data-.htm

[14] **COLMAP** – GitHub:

https://colmap.github.io/

[15] **Blender** – Fuente:

https://www.blender.org/

[16] **SQLite** – Fuente:

https://sqlite.org/index.html

[17] **Python** – Fuente:

https://www.python.org/

[18] **OpenStreetMap** – Fuente:

https://www.openstreetmap.org

[19] **BlenderGIS** – GitHub:

https://github.com/domlysz/BlenderGIS

[20] **Unity3D** – Fuente:

https://unity3d.com

[21] **Unity3D Documentation** – Fuente:

https://docs.unity3d.com/Manual/index.html

[22] **Oculus** – Fuente:

https://www.oculus.com/

[23] **Oculus Developers Documentation** – Fuente:

https://developer.oculus.com/

[24] **18 High Resolution Wall Textures** – Unity Asset Store:

https://assetstore.unity.com/packages/2d/textures-materials/brick/18-high-resolution-wall-textures-12567

[25] **POLIIGON** – Fuente:

https://www.poliigon.com/

[26] **Roof textures** – Unity Asset Store:

https://assetstore.unity.com/packages/2d/textures-materials/roof-textures-5844

[27] **textures.com** – Fuente:

https://www.textures.com/

[28] **Classic Skybox** – Unity Aset Store:

https://assetstore.unity.com/packages/2d/textures-materials/sky/classic-skybox-24923

[29] **freesound** – Fuente:

https://freesound.org/