TRABAJO FIN DE GRADO

GRADO EN INGENIERIA INFORMATICA

EditAR (working tittle)

Editor de escenas de Realidad Aumentada

Autor
Pablo Millán Cubero

Director
Francisco Javier Melero Rus

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Granada, Julio de 2023

EditAR
Editor 3D web para escenas de Realidad Aumentada

Pablo Millán Cubero

Palabras clave: *software libre, informática gráfica, Typescript, React, Firebase, NodeJS, Express, Android Studio, Kotlin, Sceneview, GLB*

Resumen

Las tecnologías de Realidad Aumentada nos permiten combinar elementos virtuales y reales en tiempo real pudiendo así crear experiencias aplicables a muchos campos; filtros con maquillaje para fotos en redes sociales, previsualizar cómo quedaría un mueble en tu propio salón, dibujar en un partido de fútbol la trayectoria que recorrió la pelota en una repetición, o incluso en videojuegos como Pokemon GO.

Con el avance de los teléfonos móviles de todas las gamas en cuanto a cámaras y capacidad de procesamiento en la última década estas experiencias han pasado a estar al alcance de todo el mundo que posea un dispositivo básico con resultados muy vistosos. Sin embargo en la mayoría de experiencias de Realidad Aumentada el usuario ocupa el rol de consumidor pasivo del contenido que generan los desarrolladores.

Se propone entonces desarrollar una aplicación web en el que el usuario pueda cargar múltiples modelos 3D con textura y animaciones, aplicarles a estos transformaciones como translaciones, rotaciones y escalado, reproducir animaciones y cargar pistas de audio. Todo ello desde un sencillo e intuitivo editor que puede usar cualquiera en el que no haga falta tener conocimientos previos. La escena 3D que se cree podrá posteriormente descargarse o guardarse en un servidor web asociado a una cuenta de usuario. El usuario podrá además, desde una aplicación Android cargar sus escenas creadas y reproducirlas, pudiendo visualizarlas en su entorno a través de la cámara del dispositivo móvil.

EditAR
Web editor for Augmented Reality scenes

Pablo Millán Cubero

Keywords: *software libre, informática gráfica, Typescript, React, Firebase, NodeJS, Express, Android Studio, Kotlin, Sceneview, GLB*

Abstract

Augmented Reality technologies let us merge virtual elements with camera footage in real time, letting us create experiences for many fields; makeup filters for social network photos, preview new furniture for your livingroom, draw the trayectory of the ball in a football match replay, or even in videogames like Pokemon GO.

With the improvement of smartphone's cameras and processing power in the last decade, this experiences are know within everybody's reach if you have a basic device. However, the mayority of Augmented Reality experiences are based in the pasive comsumption of content pregenerated by developers.

The proposed web application lets the user load multiple 3D models with textures and animations, apply transformations like translations, rotations and scales, and play audio tracks. All this in an easy and intuitive editor that anyone can use without previous knowledge. The resultant 3D scene can be either downloaded or uploaded to a web server asociated to an user account. Aditionaly, the user can load the created scenes within an Android app and play them with their device's camera, placing it in their environment.

D. **Tutora/e(s)**, Profesor(a) del ...

Informo:

Que el presente trabajo, titulado ***Chief***, ha sido realizado bajo mi supervisión por **Estudiante**, y autorizo la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a Junio de 2018.

El/la director(a)/es:

(nombre completo tutor/a/es)

Pablo Millán Cubero

Agradecimientos

A mis padres por permitirme estudiar esta carrera en Granada y apoyarme.

A mis abuelos, en especial a mi abuela Elena, que tanto presume de "su nieto el ingeniero".

A mis tíos "los franceses".

A mis amigos del Krustáceo Krujiente, Juan, Antonio, y mis compañeros de trinchera Pepe, Finn, Moisés y Pablo.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y fundamentos

En este capítulo se dará una primera introducción a la *Realiad Aumentada* y a la idea del proyecto, además de describir una serie de conceptos y tecnologías fundamentales para entender el trabajo y seguir con la lectura del resto del documento.

1.1. Descripción general y motivación

El término de Realidad Aumentada comprende al conjunto de técnicas y tecnologías que permiten la superposición e interacción de elementos virtuales sobre la realidad física a través de un dispositivo electrónico. Esta tecnología se ha aplicado a una gran variedad de ámbitos, tales como en la creación de filtros faciales para fotos en redes sociales como Instagram, dibujar la trayectoria que ha seguido el balón en la repetición de una jugada de fútbol, muestra de distintos gráficos y maquetas en los telediarios, previsualización de mobiliario en una habitación e incluso videojuegos como Pokemon GO.

El avance en las cámaras y procesadores de los teléfonos móviles en la última década ha facilitado que casi cualquier persona tenga en el bolsillo un dispositivo capaz de reproducir experiencias muy vistosas de Realidad Aumentada. Sin embargo en la mayoría de casos se trata de aplicaciones en las que el usuario consume pasivamente contenido pregenerado por los desarrolladores, sin la capacidad de crear ellos mismos estas experiencias.

Se propone entonces desarrollar en primer lugar una aplicación web que permita al usuario la composición de textitescenas 3D para uso en Realidad Aumentada. Cuando hablamos de escena nos referimos una composición de uno o más modelos 3D con una disposición concreta. Por ejemplo podríamos colocar el modelo de un tobogán junto al de un columpio y un balancín, formando el

conjunto la escena de un parque. La aplicación consistirá principalmente de un editor sencillo e intuitivo que permita cargar modelos 3D con texturas y aplicarles transformaciones para colocarlos en una escena. Adicionalmente se podrá añadir a la escena una pista de audio y reproducir animaciones de los modelos. Se podrá guardar la escena en un servidor web en una colección de escenas creadas asociadas a una cuenta de usuario, desde la cual se podrán modificar o eliminar.

Adicionalmente, el usuario podrá iniciar sesión en una app móvil Android para acceder a su lista de escenas creadas y podrá reproducir cualquiera de ellas en Realidad Aumentad con su dispositivo móvil.

Las principales motivaciones para elegir este como mi trabajo de fin de grado fue mi alto grado de interés por los gráficos 3D, mis conocimientos previos en tecnologías que acabaría usando como three.js o Android Studio, y que es un proyecto relativamente multidisciplinar, requiriendo del desarrollo de una aplicación web, una aplicación Android y un servidor web que los conecte, creando así un pequeño ecosistema y con suerte dándome una intuición de lo que puede ser realizar un proyecto parecido en un entorno laboral real.

1.2. Fundamentos

A continuación se van a describir una serie de conceptos y tecnologías esenciales para la correcta comprensión del proyecto.

1.2.1. Realidad Aumentada

La *Realidad Aumentada* (*Augmented Reality* en inglés, *RA* de aquí en adelante) hace referencia al conjunto de técnicas y tecnologías que permiten crear un experiencia en la que se combina el mundo real con contenido generado por computadores. Aunque el uso más extendido es únicamente visual y auditorio, puede apelar a más sentidos como el háptico, somatosensorial y olfativo. Se puede decir que un sistema *RA* debe incorporar tres elementos básicos: una combinación del mundo real y el virtual, interacción en tiempo real y un correcto registro de la posición y movimientos de objetos tanto reales como virtuales.

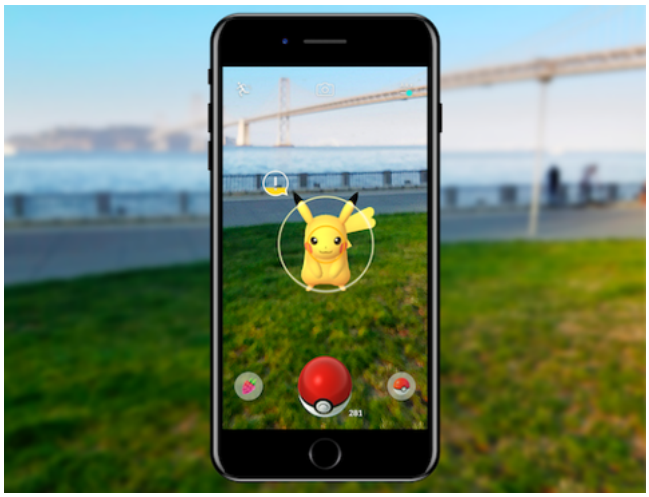

Figura 1.1: Ejemplo de Realidad Aumentada en el videojuego Pokémon Go

Existen otros términos acuñados que son a grandes rasgos sinónimos de la *RA* como *Mixed Reality*. No debe confundirse con la *Realidad Virtual*, otro ámbito en el que el objetivo es sumergir al usuario en un mundo virtual enajenándolo de sus sentidos con el mundo real, usualmente con unas gafas envolventes diseñadas especificamente para realidad virtual. Todos estas disciplinas están recogidas bajo el término paraguas de la *Extended Reality* (*XR*).

1.2.2. Visión por computador

Javier González Jiménez define en su libro *Visión por computador* [7] este campo con las siguiente cita:

"La visión por computador, también denominada visión artificial, puede definirse por el proceso de extracción de información del mundo físico a partir de imágenes, utilizando para ello un computador. Desde un punto de vista más ingenieril, un sistema de visión por computador es un sistema autónomo que realiza algunas de las tareas que el sistema de visión humano [...] realiza. La información o tareas que este sistema de visión puede llegar a extraer o realizar puede ir desde la simple detección de objetos sencillos en una imagen hasta ***la interpretación tridimensional de complicadas escenas****. "*

"[...] la información visual consiste en energía luminosa procedente del entorno. Para utilizar esa información es preciso transformarla a un formato susceptible de ser procesado [...] esta misma tarea se realiza con una cámara de vídeo (o algún otro dispositivo similar), que convierte la energía luminosa en corriente eléctrica, que puede ser entonces muestreada y digitalizada para su pprocesamiento en un computador."

En nuestro caso nos interesa especialmente la sentencia destacada de la cita. Esta disciplina nos permite extraer una idea de la estructura física de una escena, permitiéndonos detectar elementos como superficies o esquinas, indispensables si queremos por ejemplo, posicionar un objeto 3D encima de una mesa, y que el objeto parezca que siga apoyado en la mesa si movemos la cámara de lugar, cambiamos el ángulo, o nos alejamos.

Por fortuna la obtención de información a través de imágenes será un proceso completamente transparente para nosotros gracias a las distintas herramientas y tecnologías que ofrecen una API para hacer llamada de estas funciones sin que tengamos que preocuparnos por los detalles técnicos como veremos a continuación.

1.2.3. ARCore

ARCore [4] es un *SDK* (*software development kit*) de Realidad Aumentada desarrollado por Google. Ofrece una API con la que se pueden desarrollar experiencias *AR* en Android, iOS, Unity y Web renderizando modelos 3D con *OpenGL*. Para ello ARCore realiza una serie de tareas a bajo nivel relativas a la visión por computador y por las cuales el usuario no tiene que preocuparse, solo hacer las llamadas necesarias a su API. Estas funciones se detallan a continuación.

- **Seguimiento de movimiento**: Se emplea un proceso llamado *localización y asignación simultáneos*, o ANSM, para que el dispositivo obtenga un contexto sobre el mundo físico que lo rodea. Para ello ARCore detecta características visualmente distintas en las imágenes capturadas (también llamadas **puntos de atributo**) y los usa para calcular el cambio de ubicación del dispositivo. La información visual se combina con la IMU[1] del dispositivo para calcular la pose, es decir, la posición y orientación de la cámara respecto al mundo a lo largo del tiempo.

- **Comprensión ambiental**: Búsqueda de clústeres de puntos de atributos que aparentan pertenecer a superficies horizontales o verticales como pareces o mesas, y las pone a disposición del usuario como *planos* que pueden usarse para posicionar objetos.

- **Comprensión de profundidad**: ARCore detecta y almacena *mapas de profundidad* con información sobre la distancia entre dos puntos cualquiera pertenecientes a una superficie. Esto puede emplearse para simular interacciones físicas realistas entre objetos virtuales y reales o lograr el efecto de que un objeto virtual aparezca tapado o detrás de uno real.

[1] Un IMU (Unidad de Medida Inercial) es un dispositivo capaz de estimar y reportar información acerca de la velocidad y orientación del mismo a través de acelerómetros y giroscopios.

- **Estimación de luz**: Detecta información sobre la luz de su entorno, su intensidad y color. Con esto se pueden lograr efectos como que un objeto virtual a la sombra se represente más oscuro que uno expuesto a una fuente de luz.

- **Estimación de luz**: Detecta información sobre la luz de su entorno, su intensidad y color. Con esto se pueden lograr efectos como que un objeto virtual a la sombra se represente más oscuro que uno expuesto a una fuente de luz.

- **Interacción de usuario**: A través de raycasts[2], ARCore puede obtener un *punto de atributo* o modelo posicionado en la escena a partir de un gesto de usuario como tocar la pantalla táctil del dispositivo, lo que permite interacciones como colocar o mover objetos 3D.

- **Puntos orientados**: Los puntos orientados permiten posicionar objetos 3D en superficies que no son superficies planas. ARCore analiza los puntos de atributo vecinos al seleccionado para posicionar el objeto y realizará una estimación de la inclinación, devolviendo un ángulo que servirá para aplicar una pose al objeto para que se vea lo más natural posible.

- **Imágenes aumentadas**: ARCore puede detectar a través de la cámara imágenes 2D definidas previamente y asociar a cada una uno o varios modelos 3D que posicionar sobre estas en el caso de que se enfoquen. Los modelos mantendrán su posición de forma consistente si la cámara se mueve o rota.

1.2.4. SceneView

Sceneform es un *SDK* desarrollado por *Google* que permite importar y visualizar modelos 3D en distintos formatos para renderizar escenas 3D para aplicaciones de *ARCore* o realidad virtual sin necesidad de programar código *OpenGL* de bajo nivel. Cuenta con un *grafo de escena* a través del cual podemos definir la estructura de los modelos de la misma a través de un árbol de nodos, y un motor de físicas de *Filamente*[3].

Lamentablemente *Sceneform* fue descontinuado por *Google*, pero el proyecto ha renacido bajo el nombre de **Sceneview** [6], desligándose de *Google*, con código abierto y mantenido por su comunidad.

[2] text raycast

[3] Filament: Motor de físicas para escenas 3D en tiempo real desarrollado por Google.

1.2.5. Android

Android es un sistema operativo desarrollado por Google, de código abierto y basado en el núcleo de *Linux*. En un principio se diseñó con dispositivos táctiles en mente como teléfonos inteligentes y *tablets*. Es el que actualmente tiene la mayor cuota de mercado en dispositivos de este tipo, contando en 2022 con el 72,2%[4]. También se emplea en otro tipo de sistemas como televisiones y relojes inteligentes o incluso automóviles.

1.2.5.1. Android Studio

Android Studio es un entorno de desarrollo integrado o *IDE*, es decir, una aplicación informática que proporciona servicios y recursos para el desarrollador de software como un editor de código, depurador, compilador e interfaz gráfica. En este caso para el desarrollo de aplicaciones en dispositivos Android. Este tipo de herramientas es de gran interés para encarar un desarrollo ya que aglutina todas las herramientas que podemos necesitar como programadores en un solo entorno.

1.2.5.2. Kotlin

Kotlin es un lenguaje de programación multiplataforma, estáticamente tipado, con inferencia de tipos y de alto nivel. Está diseñado para ser totalmente interoperable con *Java*, pero usando una sintaxis más concisa. Es además, el lenguaje preferido por Google para el desarrollo de aplicaciones Android y el que se recomienda en toda la documentación.

1.2.6. Typescript

JavaScript (*JS*) es un lenguaje de programación interpretado y compilado *just-in-time*, aunque es más conocido como un lenguaje de *scripting*, es decir, un lenguaje que nos permite incrustar código dentro de páginas web.

Por otro lado, *TypeScript* (*TS*) es un lenguaje construido por encima de JavaScript que añade sintáxis adicional que lo hace **fuertemente tipado**. Es decir, siempre que declaremos una variable debemos especificar su tipo. Si más adelante en el código se intenta asignar un valor no correspondiente a ese tipo, el editor podrá detectarlo y avisar, pudiendo así identificar posibles errores antes siquiera de la ejecución del programa, cosa que con *JavaScript* no es posible.

[4] https://root-nation.com/es/noticias-es/es-cuota-mercado-android-ios-estadisticas-publicadas-

1.2.7. Archivo JSON

Los archivos JSON (o *JavaScript* Object Notation) es un formato de texto sencillo para el intercambio de datos. Toma como referencia la notación de objetos de *JS*, pero a lo largo de los años y debido a su simplicidad, se ha convertido en la opción por defecto para intercambio de información entre servicios web, en contraposición a su principal alternativa, *XML*.

1.2.8. React.js

React es una librería de *JavaScript* para construir interfaces de usuario web de código abierto y mantenido por su comunidad. Opcionalmente utiliza ficheros de extensión *.jsx* (*JavaScript Syntax Extension*) para facilitar el desarrollo. Estos son, como su nombre indica, una extensión sintáctica de *JS* que permite insertar código HTML, facilitando así la construcción de interfaces de usuario.

1.2.9. Node.js

Node [3] es un entorno de ejecución de *JavaScript* orientado a eventos asíncronos diseñado para crear aplicaciones de red rápidas capaces de manipular grandes cantidades de conexiones concurrentes con una alta escalabilidad para los proyectos.

1.2.10. Express

Express [1] es un *"Web application framework minimalista y flexible para Node.js que provee una serie de herramientas para aplicaciones web y móviles."* como lo definen ellos mismos. Cuenta con una gran variedad de utilidades para llamadas *HTTP* y *middleware*. Cuando hablamos de *middleware* nos referimos a un software a través del cual distintas aplicaciones se comunitan entre si por internet.

1.2.11. Firebase

Firebase [5] es un BAAS[5] desarrollado por *Google* que pone a disposición de los desarrolladores una serie de herramientas y servicios para facilitar la creación de aplicaciones. Algunas de sus servicios más importantes son:

[5] Backend-as-a-service: aplicaciones puestas al servicio de desarrolladores para que estos hagan usos de sus servicios y se centren en desarrollar el frontend de sus apps, siendo innecesario así que creen ni mantengan sus propios backends.

- **Autenticación**: Soporta autenticación de usuarios usando contraseñas con correo electrónico, número de teléfono, o cuentas de *Google*, *Facebook* y *Twitter* entre otros. Ofrece también encriptación por defecto en la base de datos para contraseñas.
- **Base de datos en tiempo real**: *Firebase* tiene también una base de datos **no relacional** que se actualiza a tiempo real y se mantiene en linea incluso cuando la aplicación no se está ejecutando.
- **Hosting**: Permite *hostear* aplicaciones web de forma fácil y rápida, pudiendo almacenar contenido en una memoria caché y accesible desde cualquier parte del mundo.

1.2.12. Archivo GLB

GLB (*.glb*) es un formato de archivo binario estandarizado para representar datos de objetos 3D como modelos, escenas, iluminación, materiales, jerarquías y animaciones. Sus siglas significan *GL Transsmission Format Binary File*. Fue introducido en 2015 como una versión binaria de los ficheros **GLTF** (*.gltf*), un formato basado en la notación *JSON* muy usado en aplicaciones web y móvil debido a su ligero peso.

Capítulo 2

Descripción del problema

Pablo Millán Cubero

Capítulo 3

Estado de la Realidad Aumentada

El software libre y sus licencias [2] ha permitido llevar a cabo una expansión del aprendizaje de la informática sin precedentes.

Capítulo 4

Planificación

4.1. Metodología utilizada

4.1.1. Metodologías de desarrollo ágil

El software es un sistema complejo de crear que depende de muchas partes interconectadas. No es igual de complejo programar un pequeño *script* que ordene los ficheros de un directorio que construir una aplicación de cierta envergadura que responda a las necesidades de un usuario o un cliente concreto, mucho más si además se debe coordinar un equipo de personas en la concepción del mismo. Existe un gran factor de riesgo e incertidumbre, sobre todo al inicio de un proyecto. Las metodologías ágiles son una forma de desarrollar software enfocada a equipos pequeños que busca paliar estos problemas a la hora de encarar un proyecto. El movimiento ágil comenzó oficialmente en 2001 con la creación del Manifiesto Ágil. Este escrito fue firmado por 17 autores y daban nombre al método que ellos usaban para desarrollar software junto con una serie de valores por los que se regían, como por ejemplo que valoraban:

- Individuos e interacciones por encima de procesos y herramientas.
- Software funcional por encima de documentación detallada.
- Colaboración con el cliente en lugar de negociación de contratos.
- Responder a cambios mientras se sigue un plan.

4.2. Temporización

4.3. Seguimiento del desarrollo

Capítulo 5

Análisis del problema

Pablo Millán Cubero

Capítulo 6

Implementación

La implementación del software se ha dividido en hitos. Estos, han sido definidos en Github y cada uno de ellos contiene un grupo de *issues* que se corresponden con las distintas mejoras que se han ido incorporando al software a lo largo de su desarrollo.

Capítulo 7

Conclusiones y trabajos futuros

Bibliografía

[1] Express.js. Fast, unopinionated, minimalist web framework for node.js. http://expressjs.com/.

[2] Free Software Foundation. GNU General Public License. http://www.gnu.org/licenses/gpl.html.

[3] Node.js Foundation. Node.js. https://nodejs.org/es/.

[4] Google. Arcore. https://developers.google.com/ar?hl=es-419.

[5] Google. Firebase. https://firebase.google.com/?hl=es-419.

[6] Thomas Gorisse. Sceneview, 3d and ar view for android, flutter and react native working with arcore and google filament. https://sceneview.github.io/.

[7] Javier González Jiménez. Visión por computador. .