Realidad aumentada sin marcadores: posibilidades, librerías y prueba de concepto.

Por
Colin Ulrich Cop, Patricia Cabrero Villar, David González Jiménez

UNIVERSIDAD COMPLUTENSE
MADRID

Grado en Grado en Desarrollo de Videojuegos
Facultad de Informática

Pedro Pablo Gómez Martín
Realidad aumentada sin marcadores: posibilidades, librerías y prueba de concepto.

Madrid, 2018–2019

Autorización de difusión y uso

Sobre TEFLON

Teflon(cc0 1.0(documentación) MIT(código))es una plantilla de LaTeX creada por David Pacios Izquierdo con fecha de Enero de 2018. Con atribuciones de uso CC0.

Esta plantilla fue desarrollada para facilitar la creación de documentación profesional para Trabajos de Fin de Grado o Trabajos de Fin de Máster. La versión usada es la 1.3.

V:1.3 Overleaf V2 with pdfLaTeX, margin 1in, NO-bib

Contacto
Autor: David Pacios Izquiero
Correo: dpacios@ucm.es
ASCII: asciifdi@gmail.com
Despacho 110 - Facultad de Informática

Índice general

Página

Resumen

Abstract

Capítulo 1

Introducción

1.1. Motivación

En los últimos años la realidad aumentada se ha convertido en una tecnología madura y presente en la mayoría de los usuarios. Apoyada por el auge de dispositivos móviles inteligentes y la mejora de los componentes de estos la realidad aumentada es adoptada cada vez más por un mayor público. Se trata de un tema de actualidad recurrente y de referencia en múltiples productos de innovación. Se estima que el tamaño del mercado

de la RA crecerá de 3.5 mil millones en el 2017 a más de 198 mil millones de dólares en el 2025.[18] En los años próximos se espera que revolucione mercados como son el arte, la educación, la publicidad, procesos de fabricación y montaje, turismo y especialmente el mundo de los videojuegos entre otros. Debido a este gran crecimiento del sector es un excelente tema para tratar de cara a conocer las limitaciones y puntos a destacar de cada una de las tecnologías existentes en la actualidad.

1.2. Objetivos

El objetivo principal del desarrollo de este proyecto será conocer las posibilidades y limitaciones de las librerías de realidad aumentada sin marcadores existentes en la actualidad. En base a este objetivo se fijaron los siguientes objetivos específicos:

a). Investigación de las principales librerías de realidad aumentada sin marcadores.

b). Implementación de pruebas de concepto de cada una de ellas para analizar los pros y contras.

c). Análisis de los resultados obtenidos de las pruebas de concepto.

d). Planteamiento e implementación de diferentes aplicaciones de realidad aumentada sin marcadores en función de los resultados obtenidos en el análisis.

e). Desarrollar una aplicación/concepto de realidad aumentada en multijugador con los cloud anchors.

1.3. Metodología

Para llevar a cabo estos objetivos se investigará a través de fuentes en internet, artículos científicos, estudios previos y libros, todos ellos reflejados en la bibliografía y webgrafía. Estos recursos serán la base de la fundamentación del proyecto y por ello se abrirán dos vías de investigación principales: se analizará por una parte acerca de las diferentes librerías de desarrollo en realidad aumentada y por otra las diferentes aplicaciones en el mercado de esta tecnología.

La revisión bibliográfica que se llevará a cabo vendrá definida por las dos áreas del conocimiento que se deben investigar de cara a desarrollar el objetivo principal planteado. En el campo del conocimiento técnico de la realidad aumentada se investigará a través de bibliografía recomendada por profesores de la Facultad de Informática de la Complutense. Los libros más significativos por tratar serán "*Handbook of Augmented Reality*" de Borko Furht, "*Augmented reality games I, Understanding the Pokémon GO phenomenon*" de Vladimir Geroimenko y "*Augmented reality games II, The gamification of education, medicine and art*" de Vladimir Geroimenko.

Debido al continuo cambio que experimentan las tecnologías de realidad aumentada gran parte de la investigación se verá supeditada a artículos científicos, así como a la documentación de las distintas librerías que lideran el mercado.

Una vez completada la investigación teórica se testearán diferentes aplicaciones ya existentes con el objetivo de encontrar sus fortalezas y debilidades. A través de estas conclusiones se podrá llevar a cabo una prueba de concepto más solida y veraz evitando cometer errores anteriormente observados. Se buscará conocer las características

específicas de las librerías escogidas delimitando los pros y contras de cada una de ellas. Para identificarlas se realizará un test definido y cerrado, poniendo a prueba las diferentes librerías en el mismo dispositivo de cara a establecer una comparativa entre todas. Estas aplicaciones test se desarrollarán cuando sea posible en el entorno Unity. Por último, se

desarrollarán tres aplicaciones de mayor nivel de complejidad que nos permitan explotar las virtudes de tres librerías diferentes de cara a mostrar y fijar las conclusiones extraídas del anterior estudio.

A continuación, se detallan las tecnologías que se utilizarán a lo largo del desarrollo del proyecto.

Para la creación de las aplicaciones, se utilizará como entorno de desarrollo principalmente **Unity 2019.3**, uno de los motores de videojuegos punteros y referentes en la industria. Gracias a su versatilidad e interfaz intuitiva nos permitirá iterar rápidamente a lo largo de los test y pruebas de concepto. Este motor es uno de los escogidos por la facultad para estudiar a lo largo del grado de desarrollo de videojuegos con lo que nos resultará más cómodo y familiar, favoreciendo de nuevo la agilidad en el desarrollo.

Como IDE [1] se utilizará **Visual Studio 2019** acompañado por Visual Studio Tools

[1] IDE: Integrated Development Environment (Entorno de desarrollo integrado)

para Unity una extensión gratuita de Visual Studio que lo convierte en una completa herramienta con la que desarrollar aplicaciones y juegos multiplataforma con Unity. Esta herramienta permite la integración de Visual Studio con el editor de Unity haciendo más eficaz el desarrollo. El sistema de control de versiones escogido será **Github** ya

que estamos habituados a la plataforma y nos permite integrarlo con herramientas como Visual Studio y Microsoft Teams.

La herramienta de seguimiento de tareas y comunicación entre el equipo escogida será **Microsoft Teams** dada su versatilidad y posibilidad de añadir herramientas.

Gracias a todas estas metodologías tanto prácticas, teóricas y técnicas y a las se lograrán cumplir los objetivos propuestos.

1.4. Plan de trabajo

La primera parte del trabajo consistirá en informarnos e investigar sobre las librerías de RA sin marcadores que lideran el mercado esta parte se realizará conjuntamente por los tres. Se estudiarán las diferentes tecnologías que componen la experiencia de la RA en los dispositivos móviles para entender cómo funciona a bajo nivel y estar actualizados con la demanda del mercado de cara a poder hacer una prueba de concepto verosímil con las aplicaciones de RA actuales.

En una segunda parte, después de identificar las librerías que existen en el mercado, se probarán (las que sean posibles) teniendo en cuenta los dispositivos y plataformas soportadas por cada una de ellas, a su vez se tendrá en cuenta la existencia de una licencia gratuita o de prueba. Se desarrollarán diversas aplicaciones de carácter básico a modo de test, éstas nos permitirán poner a prueba cada una de las principales librerías de realidad aumentada sin marcadores, comprobando su eficiencia. Los test serán ejecutados en las mismas condiciones lumínicas y con el mismo dispositivo de cara a obtener una mayor precisión en la comparación. En este caso nos dividiremos el desarrollo de los test de manera equitativa.

Una vez que se hayan encontrado las librerías que mejor se ajusten a nuestras necesidades, se desarrollarán las pruebas de conceptos las siguientes pruebas de concepto:

- Montar muebles de Ikea: se mostrará el proceso de montaje de un mueble en realidad aumentada. Ayudando al usuario a montar cada una de las piezas del proceso.
- Juego multijugador usando *cloud anchors*: se desarrollará un juego en el que poder poner a prueba las tecnologías de realidad aumentada sin marcadores combinada con puntos de localización en la nube permitiendo crear un juego multijugador.
- Visualizador de modelos 3D con las gafas Aryzon: se hará uso de unas gafas de realidad aumentada tipo *cardboard* en las que podremos ver superpuesto un modelo 3D, se podrá interactuar con el modelo 3D a través del mando de Xbox permitiendo rotarlo, escalarlo y moverlo.

Capítulo 2

Antecedentes y estado del arte

2.1. Definición

Se llama realidad aumentada al conjunto de tecnologías que permite a un usuario ver contenido virtual superpuesto al mundo real mediante un dispositivo tecnológico. Las tecnologías de realidad virtual se diferencian en que sumerge al usuario dentro de un entorno completamente sintético, sin tener consciencia del mundo real que lo rodea. La Realidad Aumentada no sustituye la realidad, sino que la complementa y puede mejorar la experiencia en ciertos ámbitos.

2.2. Historia

En la década de 1950, surgió por primera vez el término realidad aumentada cuando Morton Heilig, un cinematógrafo, pensó en un prototipo de un cine que estimulara todos los sentidos del ser humano de manera efectiva. Años más tarde, concretamente en el 1962, Heilig construyó dicho prototipo, llamado Sensorama, se trataba de un cine inmersivo y novedoso que incluía funcionalidades como 3D, visión angular (actualmente conocido como IMAX), vídeo en color, sonido en estéreo, además de estimular otros sentidos con aromas, viento, y vibraciones como podemos observar en la figura 2.1.

Figura 2.1: Cartel publicitario de la máquina Sensorama

En el 1968, Ivan Sutherland, inventó el HMD (*Human Mounted Display*), siendo así, el primer sistema que permitía ver las aristas de sencillos objetos 3D (*Wireframe*) en tiempo real. Empleaba dos sistemas de *tracking* para calcular el registro de la cámara; uno mecánico y otro basado en ultrasonidos.

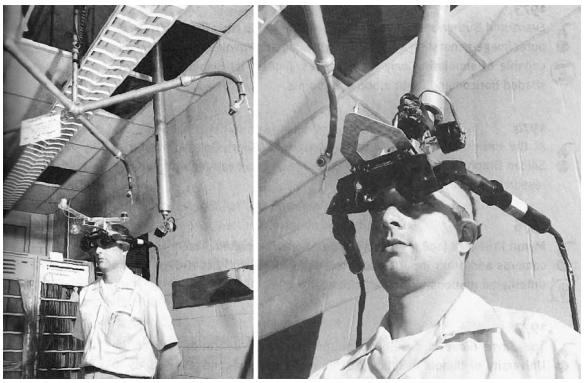

Figura 2.2: Ivan Sutherland "Espada de Damocles" 1968.

Sin embargo, no fue hasta 1992 cuando se acuñó el término de Realidad Aumentada por Tom Caudell y David Mizell, dos ingenieros de Boeing que proponían el uso de esta novedosa tecnología para mejorar la eficiencia y experiencia de las tareas realizadas por operarios humanos asociadas a la fabricación de aviones. La aparición del primer videojuego en realidad aumentada ocurrió en el año 2000. Bruce Thomas demostró en el ISWC (The International Symposium on Wearable Computers) su videojuego ARQuacke. El sistema empleaba una brújula digital, un receptor de GPS y métodos de visión basados en marcas [5]. Los jugadores tenían que llevar una especie de ordenador portátil a la espalda, un casco de visión estereoscópica y un mando de dos botones.[2]

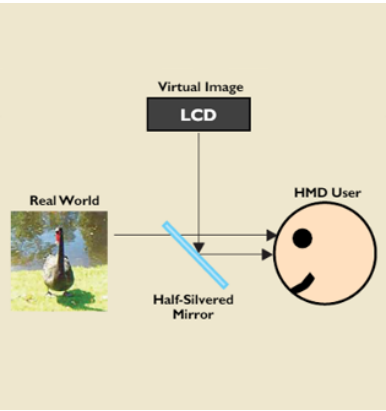

Figura 2.3: Funcionamiento del Head Mounted Display

Figura 2.4: Ejemplo de ARQuake

Figura 2.5: Dispositivo utilizado para ARQuake

2.2.1. Immersive computing

En 2007, en el ISMAR(Simposio internacional de realidad aumentada y mixta), Klein y Murray presentan el algoritmo PTAM (Parallel Tracking and Modeling) una variante al SLAM, que separa la localización y el mapeado en hilos diferentes. El SLAM es un algoritmo que sirve para que localizar la posición dentro de un entorno y modelarlo, más tarde lo explicaremos mas a fondo. Separar estos dos procesos en hilos diferentes

permitía conseguir unos resultados en tiempo real muy sólidos.

En 2008 se creó Wikitude, una aplicación que utilizaba el GPS para mostrarte información de la Wikipedia según el lugar en el que estuvieras, pudiendo aprender datos sobre monumentos, esculturas o construcciones que te fueras encontrando.

Figura 2.6: Wikitude App 2008

Un año más tarde, en 2009, desarrolló el videojuego ARrrrr del género shooter, el primer videojuego en realidad aumentada para smartphone con contenido 3D de alta calidad. Se generaba un mapa 3D sobre un marcador, y el objetivo era rescatar a los humanos de la ciudad y matar a los zombies.

Figura 2.7: ARrrrrr! Example

En el mismo año, el estudio español Novorama crea el videojuego de PSP (Portable PlayStation) Invizimals, un éxito mundial, vendiendo más de 8 millones de copias en todo el mundo en el primer trimestre de 2010. Este juego usaba la cámara addicional que se conectaba a la PSP, y con marcadores se registraba la posición del micro.

2.2.2. Project Tango

En 2014, Project Tango nació como uno de los primeros desarrollos de realidad aumentada pensado para ser distribuido mundialmente en los smartphones. En 2015, Tango pasa a formar parte de Google. El objetivo era crear un dispositivo portátil que permitiese

mapear espacios 3D. Esta tecnología solo se desarrolló para el dispositivo Phab2 Pro de Lenovo, el cual incluía un mayor número de cámaras, concretamente 3. Las tres funcionalidades principales para el desarrollo de esta tecnología han sido:

- Seguimiento del movimiento: Se trata del uso de las características visuales del entorno combinadas con los datos proporcionados por los sensores de movimiento incorporados en el teléfono, el acelerómetro y por el giroscopio, teniendo como objetivo realizar un seguimiento de los movimientos hechos del dispositivo.
- Reconocimiento del ambiente: Tango almacena la información del entorno que le rodea, buscando los puntos característicos en cada fotograma que recibe de las cámaras (más adelante explicaremos con más detalle cómo funciona el reconocimiento del ambiente).
- Percepción de profundidad: Gracias a las cámaras especiales que incorpora el dispositivo, Tango puede calcular tamaños y distancias en el entorno que se encuentra.

Después de 3 años, en 2017, Google decide cerrar el desarrollo de Tango y se centra en su tecnología actual, ARCore, que es el competidor directo de ARKit, la librería de Apple.

2.3. Aspectos técnicos

2.3.1. Descripción

La realidad aumentada es una tecnología que permite entremezclar el mundo virtual con el mundo, añadiendo la información virtual a la información física ya existente en tiempo real, permitiendo al usuario comprender mejor el entorno que le rodea. Hace unos pocos años, con la aparición de los dispositivos móviles más potentes y las librerías gratuitas de realidad aumentada, el desarrollo de esta tecnología se ha vuelto muy accesible para cualquier desarrollador. Esto implica en el número de ideas y aplicaciones que se generan cada día, que va aumentando drásticamente con el paso de los años.
En la actualidad existen tres tipos de realidad aumentada:

- Realidad Aumentada con marcadores (2D y 3D): Los marcadores pueden ser imágenes impresas o dibujos en los que la aplicación reconoce el marcador y activa la experiencia sobre dicho marcador.
- Realidad Aumentada sin marcadores: Esta es la tecnología más novedosa, ya que combina diferentes tecnologías como (SLAM, seguimiento del movimiento, reconocimiento del ambiente, detección de planos ...) para proyectar el objeto y mantenerlo en el mismo punto de anclaje sin ayuda de ningún marcador.
- Realidad Aumentada por Geolocalización: Este tipo de experiencias vinculan a la RA con una ubicación geolocalizada específica. Normalmente se utilizan en exteriores y proporcionan información contextual sobre el ambiente que nos rodea.

Las posibilidades de la realidad aumentada sin marcadores están en pleno auge y cada vez aparecen más aplicaciones que mejoran la experiencia de usuario y facilitan el trabajo en algunos sectores como puede ser la fábrica, la arquitectura, la medicina, la educación y muchos más.

2.3.2. Métodos de tracking

El Tracking es como conocemos al proceso de localización espacial del usuario en un entorno. Es uno de los aspectos clave en el desarrollo de aplicaciones de realidad aumentada ya que cuanto mejor sea la estimación de la posición y orientación del dispositivo sensor, mejores y más acertados serán los resultados y la inmersión por parte del usuario.[6]
El cálculo del *tracking* se encarga de posicionar la cámara relativamente a los objetos de la escena. Existen multitud de tecnologías y métodos para llevarlos a cabo, siendo los más comunes sensores mecánicos, magnéticos, sónicos, dinámicos y basados en visión. Éstos últimos son los más extendidos, ya que la mayoría de los dispositivos desde los que se despliegan las aplicaciones de realidad aumentada, como móviles o tablets, disponen de una o varias cámaras. [5]
El *tracking* basado en cámaras de visión es un subcampo del *tracking* 3D, en el que se utilizan algoritmos de visión por ordenador para obtener de la manera más precisa posible el posicionamiento de seis grados de libertad del dispositivo (tres grados de posición y otros tres de orientación.
En este tipo de posicionamiento es necesario disponer de un conjunto de marcadores o referencias tridimensionales para situar la cámara con respecto a ellas. Aunque recientemente se ha tendido a utilizar en menor medida los marcadores físicos para dar una experiencia más rápida y cómoda al usuario, han sido una herramienta imprescindible en los primeros pasos de la realidad aumentada para la obtención de la localización relativa de la cámara. Según David Marimón[3], fundador y director general de Catchoom, se pueden distinguir dos aproximaciones distintas a la hora del tracking: los métodos Bottom-Up y los Top-Down.

Bottom-Up

Las aproximaciones del tipo Bottom-Up pretenden obtener la posición del dispositivo basándose en la información que recibe a través de la cámara. Para este método de tracking la posición y orientación se calculan en base a la obtención de características geométricas de objetos y sus relaciones. Dependiendo de los datos procesados, el seguimiento puede ser con marcas o sin ellas.
El tracking basado en marcas era el método más extendido en los inicios de la realidad aumentada. Hace sus cálculos con la ayuda de marcadores físicos que en su mayoría presentan un gran contraste entre blanco y negro para que los sensores puedan percibirlos con mayor facilidad. Existen también marcadores que usan códigos de colores y diferentes formas geométricas, aunque después de ser sometidos a prueba se comprobó que los más sólidos eran los marcadores cuadrados. Por otra parte, este método es especialmente sensible a la oclusión, ya que cuando se pierde el marcador, es imposible calcular la posición del dispositivo. Por este motivo, se han diseñado marcadores que puedan hacer frente a este problema con imágenes en escala de grises que completan marcas que no son visibles.
Paralelamente, el *tracking* sin marcas se basa únicamente en las características intrínsecas de la escena, estructuras físicas de fácil percepción como las esquinas de una mesa.
Existen técnicas en este campo que utilizan información sobre superficies planas detectadas en el campo de visión, siendo su principal inconveniente su alto coste computacional (actualmente este tipo de localización no lo pueden llevar a cabo todos los dispositivos del mercado).

Por otra parte, hay técnicas basadas en modelos. No están considerados marcadores porque son parte del medio natural, pero al igual que con éstos los cálculos se basan en el reconocimiento de los objetos que existen y que el programa está preparado para procesar.
Finalmente, existen métodos que actúan en escenarios donde no se es capaz de obtener planos o modelos. Se suelen emplear restricciones epipolares de las cámaras en movimiento. Sin embargo, esta técnica no es utilizada habitualmente por sus altos requisitos de cómputo.

Top-Down

Las aproximaciones del tipo Top-Down intentan estimar desde la posición actual del dispositivo si se está percibiendo lo que se esperaba. Es decir, primero se estima la posición y después se confirma esa estimación con los datos del medio. En este caso, se emplean modelos del movimiento basados en filtros bayesianos para hacer una predicción de la localización del dispositivo. Partiendo de esta estimación, se busca mediante la cámara una serie de referencias parciales que corrijan la predicción y mejoren el posicionamiento del observador. Por ello, todos los modelos Top-Down se ven obligados a trabajar con filtros y modelos de asociación de datos. El uso de estos filtros permite combinar varios métodos de tracking y mantener un registro constante de los objetos y la cámara, aunque los marcadores, modelos o planos sean parcialmente visibles por oclusión o se hayan escapado del campo de visión. Además del seguimiento óptico, se han desarrollado numerosas alternativas con las que proporcionar otros métodos de localización (como los beacons o la ubicación del GPS) y así complementar y facilitar una localización más precisa y correcta. A las aproximaciones que se valen de varias de estas técnicas se las denomina métodos de fusión.

2.3.3. Tecnologías implicadas en la RA sin marcadores

El objetivo de la realidad aumentada es integrar contenido virtual en el mundo real. Idealmente, dicho contenido se tendría que comportar exactamente como uno real, esto requiere una información muy precisa sobre la posición del dispositivo que usa el usuario con respecto al objeto virtual. Para ello, se han desarrollado diferentes tecnologías que junto a los sensores de los teléfonos actuales (giroscopio, acelerómetro, sensor de luz) y a la cámara, permiten disfrutar de una experiencia casi ideal. [1]

SLAM(Simultaneuos localization and mapping)

Mapeo y localización simultáneos se le llama a la tecnología que se basa en una serie de algoritmos complejos que utiliza los datos de los sensores para construir un mapa de un entorno desconocido y a su vez para saber dónde está localizado el dispositivo. Esta técnica es usada por robots y por vehículos autónomos. ¿Cuál es el objetivo? Descubrir donde estoy. La tecnología SLAM, en el momento que empieza el algoritmo, no tiene ningún tipo de información del entorno. Normalmente, sólo tarda unos pocos segundos en crear un mapa aproximado del entorno con lo que calcula una posición inicial. Más adelante, el mapa creado va creciendo y mejorando en base a la información que obtiene desde el fotograma de la cámara. Aunque este término empezó a aparecer en la década de los 90, las primeras implementaciones carecían de cámaras o sensores que proporcionaban

información visual. En 2005 comenzó a abaratarse el coste de los ordenadores y de las cámaras, y los investigadores empezaron a combinar el SLAM con sensores visuales. Hasta este punto, esta tecnología estaba pensada para la navegación con robots en entornos desconocidos, hasta que en 2007 Georg Klein y David Murray vieron el potencial de usar esta tecnología en la realidad aumentada. [8]

Reconocimiento del ambiente

Cuanto mejor entienda la aplicación como es el entorno que le rodea, mejor será la experiencia de usuario. En este apartado, también entra el reconocimiento de superficies, tanto horizontales como verticales. El funcionamiento de esta tecnología consiste en procesar cada fotograma obtenido por la cámara y encontrar puntos característicos, estos puntos pueden ser cualquier cosa que ayuden a identificar objetos (esquinas, líneas, bordes de objetos, colores, gradientes, etc…), por lo que si se intenta detectar un plano en una superficie donde el color sea uniforme, y carezca de textura o patrones, como puede ser una pared totalmente blanca, seguramente no funcione con normalidad [16]. Con estos puntos, luego se construye una maya que va a servir como superficie en la escena de nuestra aplicación y con la cual podremos interactuar activándole las físicas y colisión.

Estimación de la luz

Esta tecnología es muy importante ya que aporta un nivel de detalle excelente, los modelos 3D virtuales se comportan como si fueran reales, se iluminan con la iluminación del mundo físico y emiten sombras. La estimación de la luz es posible gracias a la combinación de la información del seguimiento del movimiento y usando un algoritmo de análisis de imagen que determina la intensidad de la luz en la imagen del dispositivo. Cuanto más se mueva la cámara y más información recoja sobre el entorno, más precisos serán los datos de dónde viene la mayor fuente de luz, analizando el nivel de brillo de los píxeles de los fotogramas, por lo que se puede estimar la dirección en la que viene. A esta información se puede acceder desde Unity o el motor que se use, donde se le aplican los valores obtenidos a una la luz posicional.

Oclusión

El término oclusión se refiere a cuando un objeto nos impide ver otro objeto o imagen que hay detrás. Para disfrutar de una experiencia de realidad aumentada realista, esta tecnología es esencial, los objetos virtuales tienen que seguir esta regla, porque en el momento en el que cruza una persona delante del objeto, o cruzas la esquina, y sigues viendo el objeto virtual, se arruina la inmersión que podemos llegar a tener. Por lo que no sirve únicamente saber dónde está situado nuestro dispositivo con respecto al objeto, si no también hace falta saber si hay otro objeto o superficie en medio.[4]

Detección de rostros

Uno de los puntos más importantes de la realidad aumentada en la actualidad es la detección de caras y su reconocimiento.

Cada cara está compuesta por al menos 80 rasgos distinguibles, como la distancia que existe entre los extremos de la mandíbula, la profundidad de las cuencas oculares o la

separación que hay entre los agujeros de la nariz. [7]Los humanos somos especialmente buenos reconociendo estos rasgos porque tenemos una zona del cerebro dedicada específicamente a interiorizar patrones. Basándonos en el funcionamiento del cerebro de una persona hemos desarrollado algoritmos que imitan estas asociaciones, dividiendo las caras en un conjunto de puntos de referencia a los que llamamos nodal points y buscando correspondencias con otras fotos tomadas anteriormente. El algoritmo que sigue un sistema para tratar de identificar si lo que está viendo es una cara pasa por comprobar si existe en la imagen un patrón similar al que formarían normalmente los rasgos más característicos de una cara, para después preguntarse de quién es esa cara. Sin embargo, no existen dos fotos de una misma persona que sean iguales, de manera que los algoritmos tienen que lidiar con 4 problemas fundamentales a la hora de reconocer la cara de una persona: el envejecimiento, la pose, la iluminación y las emociones.

En los últimos años se ha desarrollado un sistema de reconocimiento en 3D llamado Deepface, que es capaz de tomar una foto en 2D de un individuo y crear un modelo tridimensional. De esta forma, el sistema tendrá muestras de los rasgos faciales desde todos los ángulos disponibles, solucionando así el problema de la pose.

Por otra parte, el problema del envejecimiento también ha sido debidamente prevenido, ya que al crear la estructura 3D de la cara se tienen en cuenta los nodal points más importantes y que menos varían con el transcurso de los años, como son las curvas de los ojos, de la nariz o de la barbilla. Pero lo que realmente ha supuesto un avance en este campo es el Deep Learning, que es un sistema de algoritmos que guía al programa redirigiéndole si va por mal camino. Cada vez que asocia una cara correcta o incorrectamente, registra el proceso por el que ha pasado para realizar la comprobación y queda guardado en un mapa que va ampliándose sucesivamente con cada acierto o error del sistema. De esta manera, cuantas más conexiones se creen mayor será la fiabilidad a la hora de reconocer una cara.

Facebook por ejemplo se vale de este método para el reconocimiento facial y posee una red neuronal de más de 20 millones de nodos, con una fiabilidad del 97.35 %(datos de 2015)[23] y que aun así es inferior a la capacidad de detección de una persona.

Se espera que con la mejora de la tecnología el reconocimiento facial sea una forma de identificación tan válida como las huellas dactilares y que puedan identificarse las caras de las personas incluso en grabaciones de seguridad en calidad baja y sin color, así como un método de reconocer el género, edad y otras características del individuo para ofrecerle un servicio o producto más acorde con él en ámbitos como la publicidad.

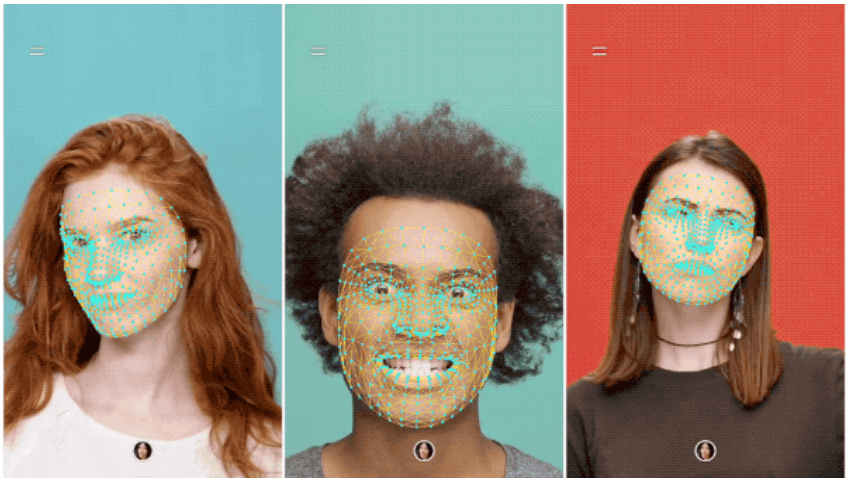

Figura 2.8: Detección de rostros

Puntos de ancla en la nube (Cloud Anchor)

El Cloud Anchor es un mecanismo que permite a los usuarios de una aplicación de realidad aumentada añadir objetos virtuales a una escena. De esta manera múltiples usuarios pueden interactuar y ver los mismos objetos desde dispositivos distintos, pero compartiendo un mismo espacio físico. Su funcionamiento es muy similar al de los anchors comunes, que se utilizan para fijar un objeto en una posición, con la diferencia de que los Cloud Anchors se hospedan en los servidores de Google. De esta manera varios dispositivos pueden consultarlos para situar los objetos en la aplicación.

Para ser utilizados, la aplicación en cuestión tiene que tener conexión a internet. Los Cloud Anchors son actualmente propios del SDK de ARCore, están soportados tanto en Android como en iOS (siempre que el dispositivo lo permita) y funcionan de la siguiente manera: ARCore tiene que generar primero un mapa de las proximidades del punto de ancla que será el centro de interés. Para ello, la cámara recopila información y características del entorno cercano desde diferentes ángulos y posiciones durante 10 segundos. Cuanto más precisa sea la información recopilada, mejor será la experiencia del usuario. Una vez transcurrido el tiempo, los parámetros del punto se hospedan en la nube y se establece el anchor, devolviendo el servidor un número de identificación único (el Cloud Anchor ID). Cuando otro usuario de la aplicación dirige su cámara hacia el mismo punto de interés, el Cloud Anchor procesa las características visuales del entorno físico desde el nuevo punto de vista. Estas características son comparadas con el mapa 3D que se ha generado anteriormente por el otro dispositivo y se establece la posición y orientación del nuevo usuario con respecto a ello para que pueda ver los objetos virtuales con la mayor precisión posible.

Para identificar un punto de ancla en la nube desde otro dispositivo se debe apuntar al lugar en que está situado sin importar la posición del dispositivo, siempre y cuando haya una línea recta entre ambos y no estén separados por una distancia superior a 10 metros.

En el caso de ARKit la tecnología para el usuario es igual, pero por dentro no funciona exactamente igual. ARKit no manda los datos a un servidor, si no que utiliza el

framework MultipeerConnectivity de Apple para mandar la información del mapa (ARWorldMap) por una conexión cliente a cliente. [22]

Cabe mencionar también que los Cloud Anchors tienen una serie de limitaciones en el almacenamiento y el acceso a los datos. Sólo pueden accederse hasta 24 horas después de haber sido colocados y 7 días después cualquier dato en la nube será borrado. El mapa hospedado en la nube no puede ser descargado por ningún usuario y no se puede determinar un lugar geográfico o reconstruir imágenes basándose en el mismo. Además, los datos que envía un dispositivo para que sean comparados con el mapa guardado no se almacenan nunca.

Para hacer un buen uso de ellos se debe evitar colocar puntos de anclado en superficies brillantes e intentar que la zona tenga una iluminación buena y consistente.

2.4. Aplicaciones

En este apartado se recogen las principales aplicaciones por sectores que existen de la realidad aumentada. Estas son la medicina, educación, arte, seguridad, publicidad, turismo, ecommerce y videojuegos.

2.4.1. Medicina

[13]

2.4.2. Educación

2.4.3. Arte

2.4.4. Fabricación

2.4.5. Publicidad

2.4.6. Turismo

2.4.7. Videojuegos

2.4.8. Comercios electrónicos(ecommerce)

Los probadores virtuales están marcando tendencia en las nuevas generaciones de aplicaciones y estrategias de venta. Estas aplicaciones hacen uso de la realidad aumentada con tecnologías como el reconocimiento de rostros, reconocimiento de superficies, estimación de luces… Siendo punto de referencia en el mercado los siguientes ejemplos.

a). **Ikea Place**
Esta aplicación permite al usuario ver el catálogo de Ikea y una vez seleccionado el elemento verlo en la habitación real con las dimensiones reales del objeto.

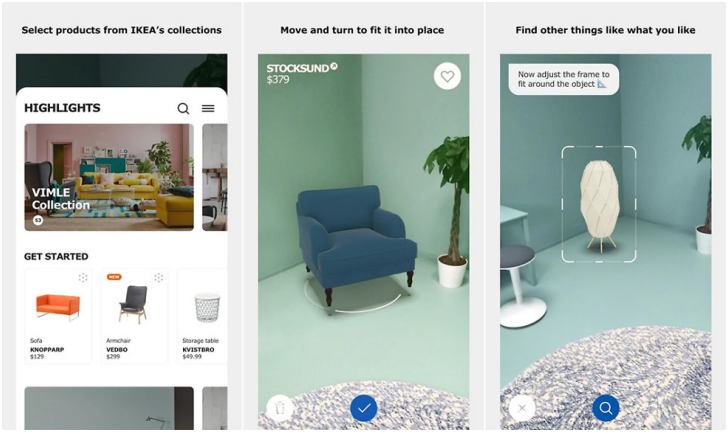

Figura 2.9: IKEA Place AR probador virtual

b). **YouCam Makeup**
Esta aplicación es un excelente y conseguido ejemplo de ecommerce en el mundo de la belleza donde se aplica esta tecnología. La calidad del tracking del pelo es bastante razonable generando una experiencia agradable.

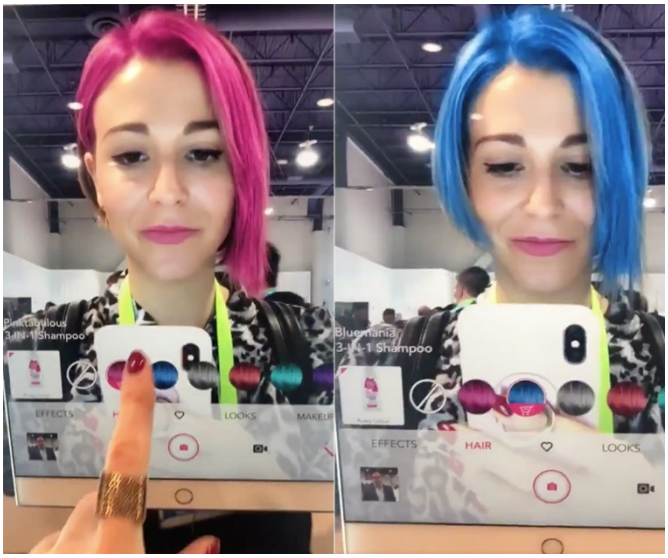

Figura 2.10: Imagen representativa de YouCam Makeup

c). **L'Oreal (Modiface)**
Aplicación que permite al usuario maquillarse con los productos de L'Oreal en realidad aumentada, así como escoger el color que más pegue con sus prendas. Mostrando los productos relacionados a esa tonalidad y ofreciendo la posibilidad de comprarlos dentro de la aplicación.

Figura 2.11: Pantallas de ejemplo de la aplicación L'Oreal Modiface

2.5. Experiencia de usuario en aplicaciones de RA sin marcadores

2.6. Librerías de realidad aumentada sin marcadores (SDK)

En este apartado se describirán las principales tecnologías de realidad aumentada sin marcadores para más tarde estudiar las capacidades y posibilidades particulares de cada una de ellas en el apartado desarrollo. Por cada librería se recogerán los siguientes datos:

- Breve descripción
- Última versión
- Funciones
- Plataformas disponibles
- Tipos de licencia

Luego se compararán todas juntas para ver las funcionalidades que tienen, las plataformas con las que son compatibles y los lenguajes que soportan.

2.6.1. Wikitude

Figura 2.12: Logo de Wikitude

Desarrollada por Wikitude GmbH, es una de las librerías pioneras en el mundo de la realidad aumentada. Lanzaron su primera aplicación en el 2008, desde entonces, son líderes del mercado. La versión que hemos usado ha sido Wikitude SDK 8.7.0 (2019-08-13).[14] Las principales funcionalidades son: Geo AR (Puntos de anclaje vía GPS) Reconocimiento de imágenes 2D (marcadores) Reconocimiento de objetos 3D Las plataformas móviles soportadas son:

- Android
- iOS
- Windows
- Unity
- Cordova
- Xamarin
- Flutter
- Titanium

Soporte para Smart Glasses:

- Epson Moverio
- Hololens
- Vuzix

Otras plataformas:

- React Native
- Ionic
- Adobe Air
- Qt by Felgo
- LBAR

Licencias:

- Wikitude Demo. Licencia de 30 días con marca de agua 499€
- Wikitude SDK PRO (Sólo con marcadores y Geo AR). 1 año de licencia 1990€
- Wikitude SDK PRO 3D (Paquete completo). 1 año de licencia 2490€

2.6.2. ARKit

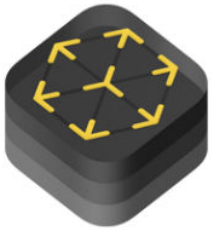

Figura 2.13: Logo de ARKit

Desarrollado por Apple, presentado en la Apple Worldwide Developers Conference de 2017. La versión con la que trabajamos es la ARKit SDK 3.0.[10] A diferencia del resto, para usar esta librería en Unity, no hace falta descargar ningún plugin, viene incluido en el paquete de Unity ARFoundation 2.2. Funcionalidades:

- Reconocimiento de imágenes 2D (marcadores)
- Reconocimiento de objetos 3D
- Reconocimiento de rostro (hasta 3 simultáneamente)
- Oclusión
- SLAM
- Estimación de luces
- Puntos de anclaje en la nube

Las plataformas soportadas son:

- iOS
- Unity (via ARFoundation)
- Unreal Engine 4.[12]

La licencia es gratuita.

2.6.3. ARCore

Figura 2.14: Logo de ARCore

Desarrollado por Google, fue lanzado en febrero de 2018 como respuesta para competir contra ARKit de iOS. La versión con la que trabajamos es ARCore SDK for Unity v1.11.0 (2019-05-05).[15] Funcionalidades:

- Reconocimiento de imágenes 2D (Marcadores)
- Reconocimiento de objetos 3D
- Reconocimiento de rostro.
- SLAM
- Mapeado de áreas grandes
- Estimación de luces
- Puntos de anclaje en la nube

Las plataformas soportadas son:

- Android
- Android NDK
- Unity (Android, iOS)
- Unreal Engine 4
- iOS

La licencia para usar ARCore es completamente gratuita.

2.6.4. Vuforia

Figura 2.15: Logo de Vuforia

Desarrollado por la empresa PTC, un proveedor tecnológico mundial de la plataforma líder de IoT (Internet of Things) y realidad aumentada. La versión que hemos utilizado ha sido Vuforia SDK Android 8.3.8 (2019-06-13).[20] Funcionalidades:

- Reconocimiento de imágenes 2D (Marcadores)
- Reconocimiento de objetos 3D
- Escáner de objetos 3D

Plataformas:

- Android
- iOS
- Windows

- Smart Glasses

Licencias:

- Básica, 42$ al mes.
- Básica con base de datos en la nube para los marcadores 99$ al mes.
- Para la versión pro, la cual incluye todas las funcionalidades, hay que contactar y hacen presupuesto a medida para la empresa.

2.6.5. Kudan

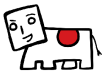

Figura 2.16: Logo de Kudan

Kudan es una empresa que se dedica al desarrollo de la realidad aumentada, virtual y mixta, además de la conducción autónoma, drones y robots. La versión que hemos utilizado ha sido la Kudan SDK Unity 1.6.0 (2019-07-16).[21] Funcionalidades:

- Reconocimiento de imágenes 2D (Marcadores)
- SLAM

Plataformas:

- Unity (Android, iOS)
- iOS
- Android

Licencias:

- AR Indie: Gratis. Pensado para la fase de desarrollo, protegido con marca de agua.
- AR Business: 1500$. Para las empresas con menos de un millón de dólares en ingresos.
- AR Enterprise: Para las empresas con más de un millón de dólares en ingresos, hay que contactar con Kudan y proporcionan un presupuesto personalizado.

2.6.6. MaxST

Figura 2.17: Logo de Maxst

Maxst se fundó en 2010 y se dedica a la investigación y desarrollo de la tecnología de realidad aumentada, han lanzado Maxst AR SDK, el cual hemos probado en la versión MaxstARSDK_Unity 4.1.3. [17] Funcionalidades:

- Reconocimiento de imágenes 2D (Marcadores)
- Reconocimiento de objetos 3D
- Reconocimiento de códigos de barras y QR.

Plataformas:

- Unity (Android,iOS)
- Android
- iOS
- Windows
- macOS
- Epson MOVERIO BT-300,350 y ODG R-7

Licencias:

- Free. Gratis, para uso no comercial, incluye marca de agua.
- Pro-one. Para aplicaciones con menos de 100k descargas (no incluye actualizaciones). Pago único de 499$
- Pro-Subscription. Subscripción anual, incluye actualizaciones 599$ por año
- Enterprise. Para aplicaciones con más de 100k de descargas. Hay que contactar con Maxst para recibir un presupuesto.

2.6.7. 8th Wall

Figura 2.18: Logo de 8th Wall

8th Wall desarrolla dos productos diferentes, 8th Wall Web y 8th Wall XR for Unity. El producto que vamos a analizar en este caso es el 8th Wall XR for Unity 11.2.6.519, para que la comparación entre las librerías sea mas precisa, ya que la potencia que tiene en navegador es menor a la que puede llegar a tener una aplicación de Unity. [9] Funcionalidades:

- Reconocimiento de imágenes 2D (Marcadores)
- 6 grados de libertad.
- SLAM
- Estimación de la luz

Plataformas:

- Unity (Android, iOS)
- Web (A-Frame, BabylonJS, Sumerian, three.js)

El uso de 8th Wall XR de Unity es gratuito. En el caso de 8th Wall Web, la licencia se cobra según las visitas en la web. Aparte, se necesita una licencia de desarrollador que cuesta 250$/mes.

Pago por visita (PPV)	Paquete estándar	PPV de alto tráfico	Paquete alto tráfico
1000$/mes	3000$/mes	6000$/mes	6000$/mes
0 visitas incluidas	500k visitas incluidas	0 visitas incluidas	5M visitas incluidas
0.01$/visita	0.01$/visita extra	0.0025$/visita	0.0025$/visita extra

Licencias 8th Wall

2.6.8. Easy AR

Figura 2.19: Logo EasyAR

EasyAR es una compañía china que lleva en el mercado desde 2016. Hemos probado la versión EasyARSense Unity SDK v3.0.1(2019-07-07)[11] Funcionalidades:

- Reconocimiento de imágenes 2D (Marcadores)
- Reconocimiento de objetos 3D
- SLAM
- Grabación de pantalla

Plataformas soportadas:

- Unity (Android, iOS)
- Android
- iOS
- Windows

Licencias:

- EasyAR SDK Basic. Gratis
- EasyAR SDK Pro. Añade el reconocimiento de objetos 3D, la grabación de pantalla y reconocimiento de más de un marcador simultáneo
- EasyAR SDK Pro trial. Lo mismo que el Pro, pero limitado a 100 usos por día.

2.6.9. ARFoundation

Figura 2.20: Logo Unity

Este último no se trata exactamente de una librería, es un paquete de Unity (aún en fase experimental) que integra una API de alto nivel (wrapper) que permite tener el mismo

código funcional para ARCore y ARKit, según si exportamos el proyecto en Android o en iOS. La versión más reciente es ARFoundation 2.2 (Unity 2019.1), la cual incluye ARKit 3.[19]
Soporta las mismas funcionalidades que ARCore y ARKit:

- Reconocimiento de imágenes 2D (marcadores)
- Reconocimiento de objetos 3D.
- Reconocimiento de rostro
- Oclusión (iOS con ARKit)
- SLAM
- Mapeado de áreas grandes (ARCore)
- Estimación de luces
- Puntos de anclaje en la nube

Plataformas soportadas:

- Unity (Android, iOS)

Su licencia, al igual que ARCore y ARKit, es gratuita.

Capítulo 3

Comparación y análisis de las librerías

El análisis de las librerías se estructurará en los puntos que se describen a continuación:

- Calidad de la documentación y primeros pasos: en este punto evaluaremos la dificultad para realizar una aplicación básica con cada librería, desde el momento en el que se descarga el SDK, hasta que se construye la APK.
- Evaluación de las capacidades de la librería: en este apartado se tendrán en cuenta las funcionalidades, las tecnologías que soporta y el nivel personalización dentro de la App, es decir, hasta que nivel podemos usar la API que nos proporcionan.
- Conclusiones: gracias al estudio realizado estableceremos unas conclusiones sobre el uso de cada librería y decidiremos si nos facilita el desarrollo de alguna prueba de concepto.

Para realizar la evaluación de las capacidades y los límites de cada librería se realizará un test que consistirá en:

- Instanciar un objeto.
- Movernos alrededor de dicho objeto, para comprobar la estabilidad del punto de anclaje.
- Realizar movimientos bruscos y veloces para ver si pierde la referencia en algún momento.
- Hacer que pierda la referencia y comprobar el tiempo en el que vuelve a aparecer el objeto.
- Alejarnos del objeto y ver hasta que distancia sigue funcionando.
- Comprobar cómo se comportan las librerías con diferentes intensidades de luz.

3.1. Wikitude

3.1.1. Calidad de la documentación y primeros pasos

3.1.2. Evaluación de las capacidades de la librería

Condiciones de luz mínimas:
Esta prueba está recogida en el repositorio: La sala está únicamente iluminada por un haz de luz perteneciente a una habitación situada al otro lado del pasillo.
Consigue instanciar el objeto dentro de un plano y posee buena iluminación. El plano falla cuando empezamos a movernos haciendo un giro de 45º no esperado, si estamos encima del modelo se pierde, si damos la vuelta completa sigue perdido. Necesitamos volver a referenciarlo por que no se recupera. La calidad del modelo y su textura es óptima para las condiciones de luz que posee. Se pierde la referencia fácilmente ante los giros. Cuando se realiza un movimiento de cámara en el que el punto de anclaje sale del *frustrum*[1] y más tarde se vuelve a enfocar a él, tarda un segundo en volver a posicionar el plano y su objeto. En esta primera prueba no hemos conseguido que pierda la referencia por distancia, se repetirá con un escenario más amplio.

	Luz Intensa	Luz Ambiente	Luz Mínimas
Distancia máxima de captura (m)			
Estabilidad del punto de anclaje			
Resistencia a movimientos			
Estimación de iluminación			
Tiempo de recuperación del ancla			

Cuadro 3.1: Análisis Wikitude

3.1.3. Conclusiones

3.2. ARKit

3.2.1. Calidad de la documentación y primeros pasos

3.2.2. Evaluación de las capacidades de la librería

3.2.3. Conclusiones

3.3. ARCore

3.3.1. Calidad de la documentación y primeros pasos

3.3.2. Evaluación de las capacidades de la librería

Condiciones de luz mínimas:

Esta prueba está recogida en el repositorio:
La sala está únicamente iluminada por un haz de luz perteneciente a una habitación

[1]Región cerrada del espacio que delimita los objetos que aparecen representados en la pantalla.

situada al otro lado del pasillo y una pequeña iluminación frontal.
Necesita más luz(2) para reconocer el plano. La estimación de la iluminación es excelente como podemos ver a lo largo de la prueba. Al realizar un movimiento alrededor del modelo no se pierde ni se desestabiliza en ningún momento. Al posicionar el teléfono sobre el modelo sigue estable. Las texturas del modelo se ven de manera nítida y realista. En la prueba de resistencia a movimientos bruscos no desaparece nunca ni vibra la imagen dando unos resultados óptimos. Cuando se realiza un movimiento en el que el modelo desaparece del campo de visión este no llega a desaparecer nunca del entorno virtual por lo que al volver a enfocar al punto de anclaje la transición es limpia.

	Luz Intensa	Luz Ambiente	Luz Mínimas
Distancia máxima de captura (m)			
Estabilidad del punto de anclaje			
Resistencia a movimientos			
Estimación de iluminación			
Tiempo de recuperación del ancla			

Cuadro 3.2: Análisis ARCore

3.3.3. Conclusiones

3.4. Vuforia

3.4.1. Calidad de la documentación y primeros pasos

3.4.2. Evaluación de las capacidades de la librería

3.4.3. Conclusiones

3.5. Kudan

3.5.1. Calidad de la documentación y primeros pasos

3.5.2. Evaluación de las capacidades de la librería

3.5.3. Conclusiones

3.6. MaxST

3.6.1. Calidad de la documentación y primeros pasos

3.6.2. Evaluación de las capacidades de la librería

3.6.3. Conclusiones

3.7. 8th Wall XR

3.7.1. Calidad de la documentación y primeros pasos

3.7.2. Evaluación de las capacidades de la librería

3.7.3. Conclusiones

3.8. Easy AR

3.8.1. Calidad de la documentación y primeros pasos

3.8.2. Evaluación de las capacidades de la librería

3.8.3. Conclusiones

3.9. ARFoundation

3.9.1. Calidad de la documentación y primeros pasos

3.9.2. Evaluación de las capacidades de la librería

3.9.3. Conclusiones

3.10. ARToolKit

Información de la librería, desarrollo de HolaMundo y conclusiones En nuestros primeros pasos en el mundo de la realidad aumentada exploramos algunas librerías como

ARToolKit, con el fin de familiarizarnos con el desarrollo de este tipo de aplicaciones. ARToolKit es una de las librerías de desarrollo pioneras en el ámbito que investigamos, disponible desde el año 2004 para descargar de manera gratuita y que cuenta con más de 160.000 descargas desde entonces. Se distribuyó para diversas plataformas como SGI IRIX (que dejó de utilizarse en 2006), Linux, MacOS y Windows y fue desarrollada originalmente por el Dr. Hirokazu Kato para posteriormente pasar a manos del Human Interface Technology Laboratory en la Universidad de Washington, la de Nueva Zelanda y ARToolworks.Inc en Seattle.

Muchas librerías posteriores se han basado en el código de ésta para ampliar sus funcionalidades, dando lugar a algunas como ARTag (que promete mayor fiabilidad a la hora de procesar imágenes por su mejor manejo de la luz), FLARToolKit (consistente en un port en ActionScript 3), ARDesktop (que facilita la creación de interfaces) o Studierstube Tracker (que mejora sus características, pero deja de ser de código abierto). Además de todas las derivaciones de ARToolKit, también podemos encontrar software no orientado a programadores como ATOMIC Authoring Tool, que permitía a cualquier usuario el desarrollo de una aplicación de realidad aumentada de manera sencilla y con una interfaz intuitiva. Esta herramienta acabó cayendo en desuso a principios de la década de 2010 debido a que ya existían librerías mejores que ARToolKit y mejores alternativas en lo que a SDK se refiere.

Al ser ARToolKit una de las primeras herramientas para el desarrollo de realidad aumentada, no contemplaba un uso de esta sin marcadores. Una de las mayores dificultades a las que se enfrentó fue el seguimiento del "ojo" del usuario, es decir, el foco de la cámara del dispositivo. Para saber desde qué perspectiva debía dibujar los elementos virtuales la aplicación necesitaba saber a dónde está mirando el usuario en el mundo real. La librería solventa este problema utilizando algoritmos de visión que calculan la localización y orientación de la cámara basándose en marcadores físicos en tiempo real. Los marcadores que es capaz de identificar consisten en la mayoría de los casos en un cuadrado negro bien contrastado con un fondo e interior blancos. Además, cada marcador, para diferenciarse del resto incluye pequeñas variaciones como otras figuras geométricas dentro del cuadrado.

Para nuestros experimentos con ARToolKit, en lugar de utilizar la librería original, utilizamos un port de la misma para ser utilizada en Unity, que puede encontrarse actualmente en https://github.com/artoolkit/arunity5. Esta extensión nos permite el acceso a componentes como ARController y ARMarker dentro del editor.

Para el desarrollo de este "HolaMundo" con ARToolKit en Unity hemos seguido los siguientes pasos: creamos un gameObject que servirá como "raíz" de la escena y otro que actuará como mánager del sistema de realidad aumentada. Al mánager le incluimos el componente ARController, que está encargado de las opciones de video y del seguimiento de los marcadores. Dentro de éste modificamos la Layer a la que debe prestar atención.

Por otra parte, el objeto raíz de la escena incluye la luz direccional y la cámara, y además le añadimos el script AROrigin, que permite situar espacialmente la escena. La cámara, además de su script de cámara recibe un ARCamera para poder detectar los marcadores.

Ahora creamos un objeto que llevará la información del marcador y le añadimos el componente ARMarker, que lleva el tag del marcador que hace las veces de identificador único. Este componente tiene dos tipos de patrones para identificar por defecto: hiro y kanji. En este caso utilizaremos el patrón "hiro", que es el que consiste en un cuadrado negro simple.

Añadimos a la raíz de la escena un objeto que será contenedor del objeto 3D que queremos que aparezca cuando enfocamos al marcador y que lleva el script ARTrackedObject y dentro del campo "Marker Tag" introducimos el identificador del marcador asociado al objeto.

Conclusiones: si bien este sistema fue útil en su día para sentar las bases del desarrollo de programas en realidad aumentada, hoy en día no tiene mucho sentido su uso. No se encuentra casi documentación actualizada para su uso y la página web que le daba soporte (artoolkit.org) ha desaparecido. Además, sus funcionalidades son muy limitadas y su rendimiento es muy inferior al que presentan otras alternativas más actuales como Vuforia, que permite también el uso de marcadores. Sí que es cierto que una aplicación generada con esta librería y que realiza la misma funcionalidad que otra, pero generada usando Vuforia pesa ligeramente menos. Sin embargo, no consideramos que sobre todo el conjunto este dato sea lo suficientemente significativo como para plantearse su uso.

3.11. Evaluación

3.11.1. Tabla de funcionalidades

A continuación, hay una tabla en el que se puede comparar rápidamente las funcionalidades que tiene cada una de las librerías que vamos a analizar. Más adelante, en la parte de comparación y análisis de las librerías, evaluaremos la calidad y eficiencia de estas funcionalidades para cada librería.

SDK	Reconocimiento 2D	Reconocimiento 3D	Detección de planos	SLAM	Reconocimiento de rostro	Estimación de luces	Otras
Wikitude	✓	✓	✓	✓	-	✓	Geo AR
ARKit	✓	✓	✓	✓	✓	✓	Oclusión, Cloud Anchor
ARCore	✓	✓	✓	✓	✓	✓	Cloud Anchor
Vuforia	✓	✓	✓	✓	-	✓	
Kudan	✓	-	✓	✓	-	✓	
MaxST	✓	✓	✓	✓	–	✓	
8th Wall XR	✓	–	✓	✓	–	✓	
EasyAR	✓	✓	✓	✓	–	✓	Grabación de pantalla
AR Foundation	✓	✓	✓	✓	✓	✓	

Cuadro 3.3: Comparación de funcionalidades

Como vemos en la tabla, menos en el reconocimiento de rostro, casi todas las librerías tienen las mismas funcionalidades, por lo que a priori nos sirve cualquiera para realizar

nuestras pruebas de concepto, pero antes toca probarlas y ver cuál es la librería que mejor implementadas y pulidas tiene estas funcionalidades.

3.11.2. Usabilidad

Este apartado consistirá en dos apartados, en la primera compararemos en que plataformas se pueden ejecutar las librerías, y luego comparemos en que plataforma y lenguajes se pueden programar las aplicaciones.

SDK	Unity3D (Android, iOS)	Unreal Engine 4	Java	Objective-C	C++	JavaScript
Wikitude	✓	–	✓	✓	–	✓
ARKit	✓	✓	–	✓	–	–
ARCore	✓	✓	✓	✓	–	–
Vuforia	✓	–	✓	✓	✓	–
Kudan	✓	–	✓	✓	–	–
MaxST	✓	–	✓	–	✓	–
8th Wall	✓	–	–	–	–	✓
EasyAR	✓	–	✓	✓	✓	–
AR Foundation	✓	–	–	–	–	–

Cuadro 3.4: Comparación de plataformas y lenguajes soportados

Podemos ver que prácticamente todas las librerías soportan Android e iOS, con lo que podemos deducir que es donde más se está invirtiendo en el mercado. Quizás las Smart Glasses puedan brindar una experiencia de realidad aumentada más agradable, pero todavía está muy lejos de ser accesible para la mayoría de la población, mientras que un dispositivo móvil es mucho más asequible.
Con esta tabla, es muy fácil ver que Unity3D sale ganador, podemos trabajar con absolutamente todas las librerías, además, gracias a su editor y entorno visual, resulta muchísimo más cómodo y ahorra mucho tiempo a la hora de hacer una aplicación de realidad aumentada.

3.12. Conclusiones prueba de concepto

	Wikitude	ARKit	ARcore	Vuforia	MaxST	EasyAR	Kudan	8th Wall XR	ARFoundation
Calidad de la documentación									
Distancia máxima de captura (m)									
Estabilidad del punto de anclaje									
Comportamiento con luz intensa									
Comportamiento con luz ambiente									
Comportamiento con luz tenue									
Estimación de luces									
Total (puntuación)									

Cuadro 3.5: Análisis de las características de las librerías de RA sin marcadores

Capítulo 4

Pruebas de concepto

4.1. Propuestas

Una vez se han testeado y analizado las diferentes tecnologías surgen diferentes ideas como prueba de concepto en función de las características particulares de cada tecnología. Las cuales son enumeradas a continuación:

- Desarrollo de un *plugin* capaz de gestionar los *cloud anchor* en interiores y almacenar la información durante un período de tiempo ajustable a las necesidades del producto. Será capaz de identificar estas posiciones *online* y *offline* mediante una base de datos almacenada tanto en servidor como en local si fuese necesario. Plataformas deseables: Android (prioritario) y iOS.
- *High level* API que permite desarrollar de manera más versátil y sencilla aplicaciones en realidad aumentada teniendo compatibilidad plena entre ARKit y ARCore. El usuario será capaz de desarrollar una aplicación en realidad aumentada completa sin apenas líneas de código ya sea mediante *blueprints* o módulos.
- Juego multijugador con *Cloud Anchors*. Juego en el que dos jugadores o más compiten por conseguir más puntos por destruir edificios. (Inspirado en Bombardero - Amstrad CPC)
- Plataforma de realidad aumentada: permitirá al usuario acceder en el momento a diferentes contenidos ya sean vídeos, juegos o experiencias sin necesidad de salir de la aplicación. Esta idea estaba pensada para el uso con unas gafas de AR y mando inalámbrico para el control del dispositivo. Inspirado en Google DayDream.
- Reconocimiento de objetos.
- Proceso de montaje de muebles paso a paso del en realidad aumentada.

4.2. Juego multijugador con cloud anchor (ARCore)

4.3. Instrucciones de montaje de muebles en AR

4.4. Visualizador de objetos en AR con gafas Aryzon

Capítulo 5

Conclusiones

Capítulo 6

Contribuciones

6.1. Colin Ulrich Cop

6.2. Patricia Cabrero Villar

6.3. David González Jiménez

Índice de figuras

Índice de cuadros

Bibliografía

[1] Julie Carmigniani y Borko Furht. "Augmented Reality: An Overview". En: 3.ª ed. Vol. 4. 5. Handbook of Augmented Reality. Nueva York: Springer, jul. de 1993. Cap. 1, págs. 03-46.

[2] Wayne Piekarski y Bruce Thomas. *ARQuake: The Outdoor Augmented Reality Gaming System*. 1.ª ed. Bubok Publishing S.L, 2002.

[3] D. Marimon. "Advances in Top-Down and Bottom-Up Approaches to Video-Based Camera Tracking". Tesis doct. France: École Poly- technique Fédérale de Lausanne., jul. de 2007.

[4] Guan Tao y Wang Cheng Wang Tian Yuan. "Real-Time Occlusion Handling in Augmented Reality Based on an Object Tracking Approach." En: *MDPI* 10.10 (jul. de 2010).

[5] J.A Albusac Jiménez y J.J Castro Sánchez C. González Morcillo. *Realidad Aumentada. Un enfoque práctico con ARToolkit y Blender*. 1.ª ed. Bubok Publishing S.L, 2012.

[6] Shoaib Ehsan y Adrian F. Clark Erkan Bostanci Nadia Kanwal Nadia. "User Tracking Methods for Augmented Reality". En: *International Journal of Computer Theory and Engineering* 5 (2013), pág. 98.

[7] BBC Earth Lab. *How Does Facial Recognition Work? | Brit Lab*. 2015. URL: https://www.youtube.com/watch?v=1aHub80AHFk (visitado 02-09-2019).

[8] Maxst. *SLAM, Core technology of AR, What is it?* 2018. URL: https://medium.com/maxst/slam-core-technology-of-ar-what-is-it-e6c9ae4839b4 (visitado 02-09-2019).

[9] 8thWall. *8thWall developer documentation*. 2019. URL: https://www.8thwall.com/index.html (visitado 01-09-2019).

[10] Apple. *Apple Developer Documentation*. 2019. URL: https://developer.apple.com/augmented-reality/arkit/ (visitado 01-09-2019).

[11] EasyAR. *EasyAR*. 2019. URL: https://www.easyar.com/view/sdk.html (visitado 01-09-2019).

[12] E. Games. *Getting started with Unreal Engine 4 and Arkit*. 2019. URL: https://www.unrealengine.com/en-US/blog/getting-started-with-ue4-and-arkit (visitado 01-09-2019).

[13] V. Geroimenko. *Augmented reality games II, The gamification of education, medicine and art*. 1.ª ed. Springer, 2019.

[14] W. GmbH. *Wikitude Documentation*. 2019. URL: https://www.wikitude.com/download/ (visitado 01-09-2019).

[15] Google. *ARCore developer documentation*. 2019. URL: https://developers.google.com/ar/ (visitado 01-09-2019).

[16] Google. *ARCore Fundamental Concepts.* 2019. URL: https://developers.google.com/ar/discover/concepts (visitado 02-09-2019).

[17] Maxst. *Maxst developer documentation*. 2019. URL: https://developer.maxst.com/ (visitado 01-09-2019).

[18] Statista. *Topic: Augmented Reality (AR)*. 2019. URL: https://www.statista.com/topics/3286/augmented-reality-ar/ (visitado 07-08-2019).

[19] Unity. *About AR Foundation*. 2019. URL: https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@2.1/manual/index.html (visitado 01-09-2019).

[20] Vuforia. *Vuforia developer documentation*. 2019. URL: https://developer.vuforia.com/pricing (visitado 01-09-2019).

[21] Xlsoft. *Kudan developer documentation*. 2019. URL: https://www.xlsoft.com/en/products/kudan/price.html (visitado 01-09-2019).

[22] Lior Wolf Yaniv Taigman Ming Yang y Marc'Aurelio Ranzato. *Creating a Multiuser AR Experience*. 2019. URL: https://developer.apple.com/documentation/arkit/creating_a_multiuser_ar_experience?language=objc (visitado 02-09-2019).

[23] Marc'Aurelio Ranzato Yaniv Taigman Ming Yang y Lior Wolf. *DeepFace: Closing the Gap to Human-Level Performance in Face Verification*. 2019. URL: https://research.fb.com/publications/deepface-closing-the-gap-to-human-level-performance-in-face-verification/ (visitado 02-09-2019).

PASCAL
ENERO 2018
Ult. actualización 8 de septiembre de 2019
LaTeX lic. LPPL & powered by TEFLON CC-ZERO