

Práctica 4:

Realidad Aumentada

con Vuforia



Máster Universitario en Ingeniería Informática
[Realidad Virtual]



❖ Índice:

1) Identificar la aplicación de Realidad Aumentada

2) Análisis del software

- a. Componentes y ejecución asociados a Vuforia
- b. Origen de los objetos de conocimiento
- c. Framework base y sistema de despliegue
- d. Gestión de inputs y outputs
- e. Escalabilidad
- f. Reutilización
- g. Acoplamiento
- h. Mantenimiento

3) Análisis de posibles problemas

- a. Visualización entre dispositivos
- b. Beneficios/alteraciones para la salud
- c. Calidad del software
- d. Privacidad

4) Referencias

1. Identificar la aplicación de Realidad Aumentada

La aplicación de Vuforia realizada desde cero por mí salvo los modelos 3D, consiste en mostrar una figura 3D sobre una imagen concreta a través de la pantalla de un dispositivo Android. En ese caso habrá dos imágenes que hacen de “Triggers”. En una imagen se mostrará un helicóptero de transporte con animación y en la otra se mostrará un Stormtrooper de Star Wars bailando.

2. Análisis del software

a. Componentes y ejecución asociados a Vuforia

En mi caso Vuforia no estaba instalado por defecto en Unity por lo que tuve que importarlo. Para ello tuve que descargar Vuforia desde el enlace [1]. Después, haciendo uso del “Package Manager” importé el archivo JSON de Vuforia. Una vez instalado Vuforia se puede comprobar que está bien instalado como aparece en la Figura 1.

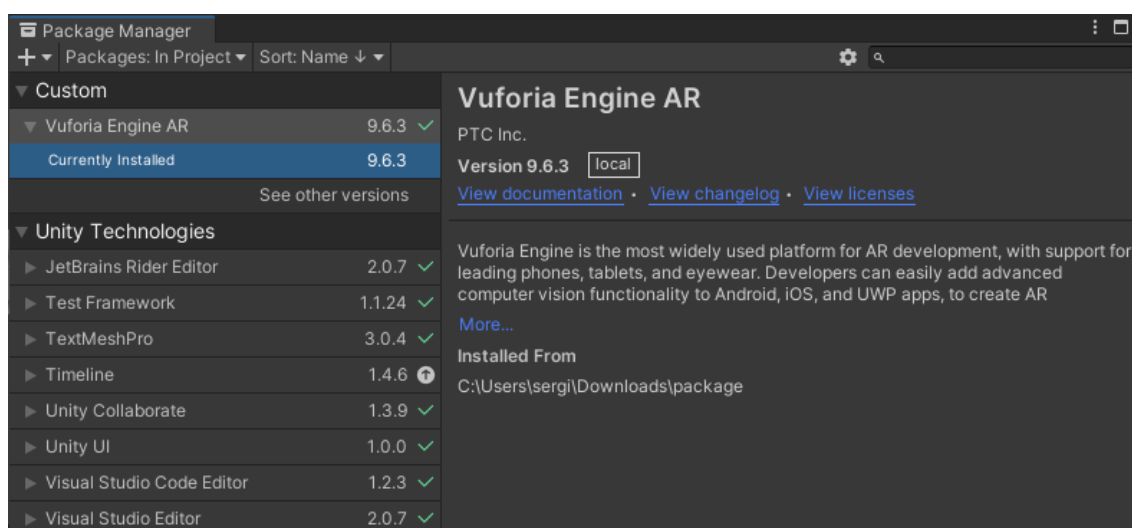


Figura 1 - Vuforia en Package Manager

Una vez realizados estos cambios, dentro de la escena inicial creo una “AR Camara”. Es importante introducir la KEY en el recurso de Vuforia para que se pueda utilizar correctamente estos elementos de realidad aumentada. El recurso se puede ver en la Figura 2.

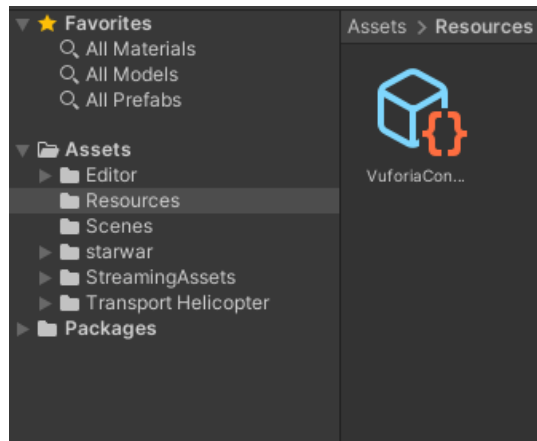


Figura 2 - Recurso de Vuforia

Ahora he creado una “Image Target” que es la imagen que se utilizará como base para mostrar las figuras 3D. He asignado una imagen que la tengo almacenada en la base de datos en mi perfil de Vuforia. No obstante, de las imágenes y de la base de datos hablaré más detalladamente en el apartado B. Tanto la KEY como la base de datos se pueden crear/ver al iniciar sesión en el portal de Vuforia[2].

Por último, se añaden los modelos 3D a la escena y se anclan a las imágenes.

b. Origen de los objetos de conocimiento

Como ya mencioné anteriormente, las imágenes que utilizo como “Triggers” para mostrar las figuras 3D, las tengo almacenadas en la base de datos en mi perfil de Vuforia. Se puede ver la base de datos en la Figura 3.

AR [Edit Name](#)
Type:

Targets (3)

Add Target

Download Database (All)




<input type="checkbox"/> Target Name	Type	Rating ⓘ	Status ▾	Date Modified
<input type="checkbox"/>  001	Single Image	★★★★★	Active	Apr 05, 2021 19:30
<input type="checkbox"/>  star_wars	Single Image	★★★★☆	Active	Apr 05, 2021 19:20
<input type="checkbox"/>  plataforma_helicoptero	Single Image	★★★★☆	Active	Apr 04, 2021 21:07

Figura 3 - Base de datos en Vuforia

Las imágenes utilizadas para mostrar las figuras son las que se muestran en la Figura 4. Se mostrarán un helicóptero volando en la de la izquierda y un Stormtrooper bailando en la de la derecha.

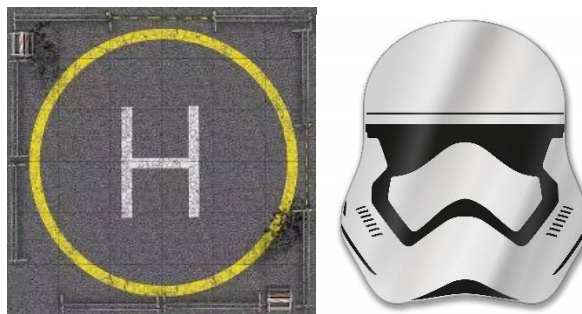


Figura 4 - Imágenes utilizadas como Triggers

c. Framework base y sistema de despliegue

El framework o entorno que he utilizado se corresponde a Unity 2020.3.2f1. La versión de Vuforia para este proyecto es la 9.6.3.

Al ser un proyecto donde el dispositivo final es un dispositivo Android, cambio la plataforma de destino. Para ello se utiliza la opción “Building Settings” y la cambio manualmente a Android. En la ventana emergente de esta opción, está el botón “Player Settings”. Este botón muestra la ventana que se puede ver en la Figura 5.

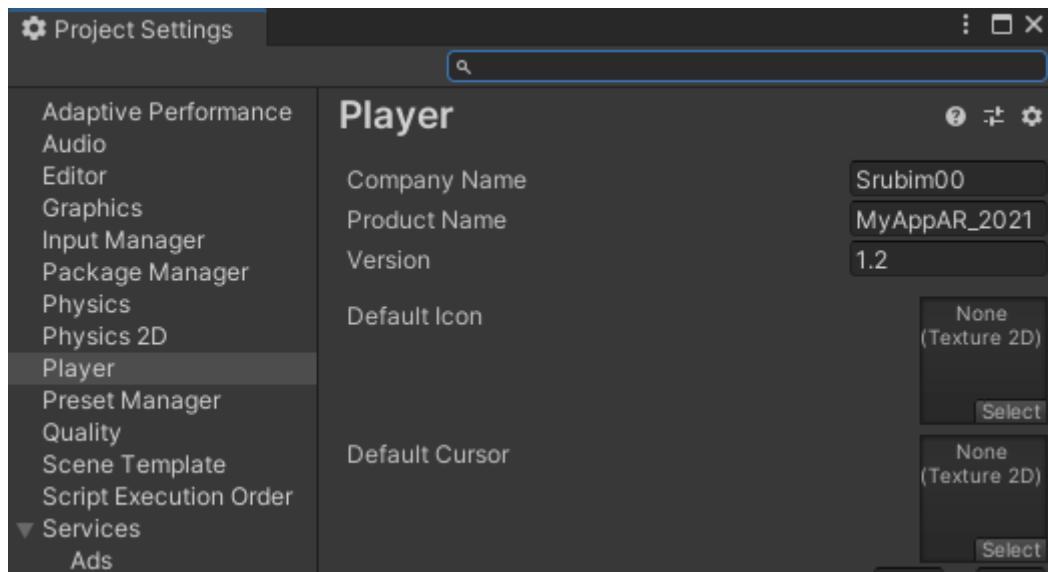


Figura 5 – Ventana Player Settings

En la ventana mostrada en la Figura 5 le doy el nombre a la aplicación, la versión de despliegue y el nombre de la APK que se generará para la instalación.

Por último, para generar el APK de la aplicación, en “Building Setting”, le doy a la opción de Build como se puede ver en la Figura 6.

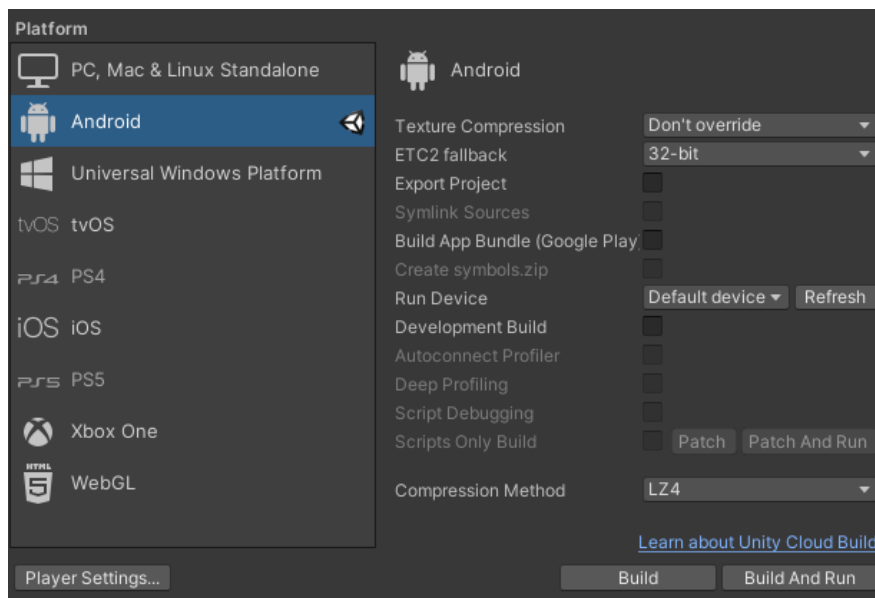


Figura 6 - Ventana Building Setting

d. Gestión de inputs y outputs

Los dispositivos físicos que trabajan como inputs serían las cámaras de los dispositivos Android, pero en cuanto a inputs a nivel de software serían todas las imágenes utilizadas como “Triggers” para mostrar las figuras 3D. En cierto modo, los modelos 3D con sus respectivas animaciones son inputs ya que los introduces dentro del proyecto.

Los outputs físicos serían las pantallas del dispositivo. En cuanto a los outputs a nivel de software sería el poder mostrar el correcto funcionamiento de las animaciones de los modelos 3D sobre las imágenes.

e. Escalabilidad

Considero que hay dos tipos de escalabilidad en este proyecto:

- Por número de figuras mostradas al mismo tiempo:
Existe un problema de escalabilidad. Mi proyecto solo puede mostrar una figura al mismo tiempo.
- Por número de imágenes reconocibles para mostrar las figuras 3D:
Alta escalabilidad ya que permite asociar diferentes figuras 3D a multitud de imágenes o fotografías. No hay constancia de un límite.

f. Reutilización

Mi aplicación no tiene problema en ser ejecutada en dispositivos Android durante un largo tiempo, posiblemente años. No obstante, si un usuario quiere realizar modificaciones en el proyecto de un tercero, seguramente tenga problemas principalmente por:

- Versión de Unity
- Versión de Vuforia

- Versión SDK
- Versión JDK
- Versión NDK

En mi repositorio de Github [3] dejaré el código de esta aplicación y este fichero PDF donde describo lo que he realizado para crear mi proyecto.

g. Acoplamiento

El conjunto de elementos del proyecto se agrupa o se acopla tal y como se muestra en la Figura 7. Se puede ver que hay un alto acoplamiento entre elementos del diseño del proyecto.

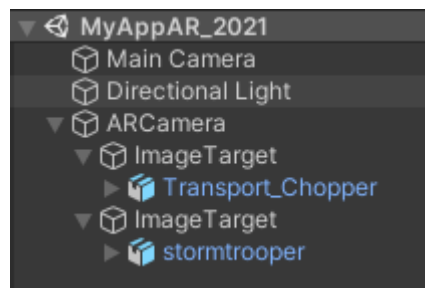


Figura 7 - Estructura de los elementos

He utilizado mi aplicación en varios dispositivos móviles Android y han funcionado correctamente por lo que no existe un alto acoplamiento respecto a los dispositivos de despliegue.

h. Mantenimiento

Me reitero en que el mantenimiento de aplicaciones de Realidad Aumentada puede llegar a ser muy complejo y tedioso debido a los problemas entre versiones que pueden surgir. El caso más palpable es mi propia experiencia a la hora de intentar hacer funcionar algún repositorio de Github. Antes de ponerme a realizar mi propia aplicación,

intenté hacer funcionar algunas aplicaciones de Github y surgieron demasiados errores de compatibilidad entre versiones.

No obstante, para mi aplicación, al tener especificado las versiones utilizadas de cada software se puede replicar la ejecución sin ningún problema salvo posibles incompatibilidades con el propio equipo de trabajo.

3. Análisis de posibles problemas

a. Visualización entre dispositivos

La aplicación se ha mostrado correctamente en dispositivos móviles Android con la versión 10 de Android. A continuación, se muestran las imágenes de la ejecución directa de la aplicación.



Figura 8 - Capturas del funcionamiento de la aplicación

b. Beneficios/Alteraciones para la salud

En cuanto a posibles alteraciones negativas en la salud que puedan ser causadas por este tipo de tecnologías, cabría decir que son casi nulas, ya que lo máximo que pueden causar es un pequeño mareo o cansancio ocular, al igual que si estuviese mirando una pantalla de un ordenador durante bastante tiempo. Por ello se destaca la realidad aumentada como una fuente de conocimiento con mucho potencial, que se puede aplicar en diversas ramas del conocimiento como puede ser la medicina y la educación entre otros.

c. Calidad del software

El lenguaje usado en las librerías de Unity mayoritariamente es C#. No obstante, al ser un proyecto donde solo se utilizan unas imágenes bases, funciones del paquete de Vuforia y texturas, no contiene C#, sino más bien mi proyecto está construido por otros archivos. Para hacer un análisis de este software y su calidad, he utilizado VS Counter que es un módulo de Visual Studio Code que permite medir el número de líneas que componen el proyecto junto con un listado de todos los archivos involucrados o que son accesibles desde el proyecto. [4] Hay un total de 1103 ficheros, donde hay 113.321 líneas de código, 18.983 comentarios y 27.089 líneas en blanco, es decir, hay un total de 159.393 líneas.

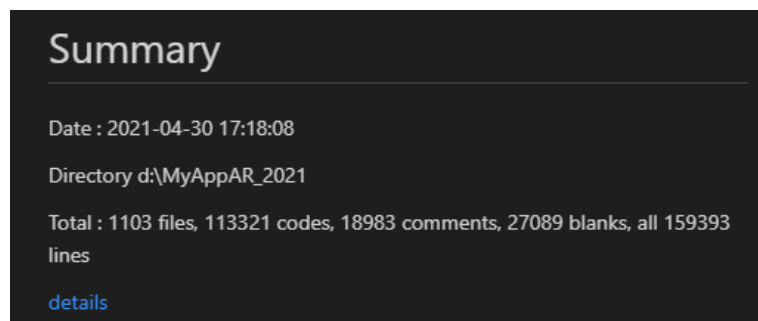


Figura 9 – Resumen del análisis del proyecto 1/3

A continuación, se mostrarán los lenguajes implicados en la totalidad del proyecto junto con paquetes que son del propio Unity. Un ejemplo claro es el de C# que es el que compone las librerías de Unity, no obstante, para el ejemplo de Vuforia que he realizado yo, no he utilizado en ningún momento la programación en C#.

language	files	code	comment	blank	total
C#	898	104,313	18,968	24,895	148,176
Markdown	160	4,646	0	2,011	6,657
YAML	3	1,812	0	4	1,816
JSON	26	1,467	0	15	1,482
Log	7	822	0	101	923
HLSL	2	164	11	44	219
ShaderLab	1	56	4	16	76
XML	6	41	0	3	44

Figura 10 – Resumen del análisis del proyecto 2/3

El análisis realizado en el propio directorio y sus respectivos subdirectorios corrobora el resultado que aparece en las figuras 8 y 9.

path	files	code	comment	blank	total
.	1,103	113,321	18,983	27,089	159,393

Figura 11 – Resumen del análisis del proyecto 3/3

d. Privacidad

Todo lo que la cámara del dispositivo Android capte no es almacenado ni guardado en algún tipo de base de datos o servidor. La cámara solo intenta encontrar la imagen que se utilizará como “Trigger” y así poder mostrar el modelo 3D asociada a la misma. En conclusión, en esta aplicación se puede afirmar que el usuario que la utilizada mantiene íntegramente su privacidad, por lo que puede estar seguro.

4. Referencias

[1] Developer.vuforia.com. n.d. *SDK Download | Vuforia Developer Portal*. [online] Available at: <<https://developer.vuforia.com/downloads/sdk>>.

[2] Developer.vuforia.com. n.d. *Vuforia Developer Portal*. [online] Available at: <<https://developer.vuforia.com/vui/auth/login>>.

[3] Rubio, S., 2021. *srubim00/MyAppAR_2021*. [online] GitHub. Available at: https://github.com/srubim00/MyAppAR_2021

[4] Marketplace.visualstudio.com. 2018. *VS Code Counter - Visual Studio Marketplace*. [online] Available at: <<https://marketplace.visualstudio.com/items?itemName=uctakeoff.vscode-counter>>.