Aplicaciones Móviles 3D: un estudio comparativo de performance y consumo de energía

Federico Cristina[1], Sebastián Dapoto[1], Pablo Thomas[1], Patricia Pesado[1], Jefferson Perez Altamirano, Martin De la Canal Erbetta

[1] Instituto de Investigación en Informática LIDI,
Universidad Nacional de La Plata – Argentina
Centro Asociado Comisión de Investigaciones Científicas de la Provincia de Buenos Aires

{fcristina, sdapoto, pthomas, ppesado}@lidi.info.unlp.edu.ar
jefferson556@gmail.com
martin_delacanal@hotmail.com

Resumen. En los últimos años la tecnología ha evolucionado exponencialmente, con avances en todos los ámbitos. Particularmente, la tecnología en dispositivos móviles se ha vuelto cada vez más sofisticada, robusta y con gran capacidad de cómputo. Debido a estos avances, actualmente es posible ejecutar aplicaciones móviles complejas con exigentes requisitos de hardware y energía, como por ejemplo las aplicaciones 3D. Existen, sin embargo, dos factores primordiales a considerar al momento de desarrollar aplicaciones móviles 3D: la performance y el consumo de energía, los cuales tienen un impacto directo en la experiencia de uso. El presente trabajo evalúa y compara estos aspectos en los dos frameworks de desarrollo 3D más utilizados actualmente: Unity y Unreal Engine.

Palabras Clave: dispositivos móviles, aplicaciones 3D, Unity, Unreal Engine, performance, consumo de energía

1 Introducción

El continuo avance en la capacidad de procesamiento de los dispositivos móviles ha permitido el desarrollo de aplicaciones cada vez más complejas, las cuales incluyen características tales como visualización 3D o realidad aumentada, características computacionalmente exigentes.

Este tipo de aplicaciones demandantes de alto requerimiento de hardware, presentan básicamente dos factores a considerar al momento del desarrollo:

- La performance, la cual tiene un impacto directo en la fluidez de la visualización y por ende en la experiencia de uso.
- El consumo de energía, que puede llegar a definir la utilización - o no - de una aplicación.

Es posible aseverar que en cierta medida ambos aspectos se encuentran en estrecha relación, dado que mayor performance demanda mayor potencia de cómputo y por

ende mayor consumo de energía. Sin embargo, es necesario considerar a estos dos factores por separado a fin de poder realizar un estudio independiente sobre las herramientas de desarrollo de aplicaciones 3D presentadas en este trabajo.

Existen diversos frameworks de desarrollo de aplicaciones 3D. Dos de los frameworks más populares actualmente son Unity [1] y Unreal Engine [2].

Unity se destaca por la cantidad de documentación disponible, una enorme y muy activa comunidad de usuarios, gran variedad de componentes pre-desarrollados (assets) y plugins que facilitan la integración con otras herramientas. Además, tiene una elevada curva de aprendizaje, dado la sencillez de su editor y lenguaje de programación requerido. Haciendo uso de los manuales y tutoriales existentes es posible simplificar el desarrollo de una aplicación móvil 3D. Por otro lado, Unity ofrece una gran cantidad de plataformas de publicación.

Unreal Engine ofrece una gran calidad y potencia de gráficos. Aunque no tan extensa como la de Unity, Unreal cuenta con una comunidad muy activa y colaborativa. Dispone además de una buena cantidad de documentación y tutoriales. En términos generales, el desarrollo de aplicaciones 3D en Unreal es más complejo que en Unity. Sin embargo, en las últimas versiones se ha incluido el sistema de Blueprints, una opción de programación basada en nodos y componentes que facilita el trabajo de codificación.

El presente estudio tiene como finalidad analizar y comparar la performance y el consumo de energía independientemente sobre estos dos frameworks de desarrollo de aplicaciones 3D. El trabajo se organiza del siguiente modo, en el capítulo 2 se presenta la motivación, el capítulo 3 muestra una propuesta de evaluación, el capítulo 4 detalla el desarrollo de los prototipos, el capítulo 5 muestra las pruebas efectuadas y presenta los resultados obtenidos, en el capítulo 6 se presentan las conclusiones y trabajo a futuro.

2 Motivación

El continuo avance en la capacidad de procesamiento de los dispositivos móviles ha permitido el desarrollo de aplicaciones cada vez más complejas, las cuales incluyen características tales como visualización 3D o realidad aumentada, características computacionalmente exigentes.

Este tipo de aplicaciones demandantes de alto requerimiento de hardware, presentan básicamente dos factores a considerar al momento del desarrollo:

El análisis de performance se inició durante el proceso de desarrollo de aplicaciones móviles 3D inmersivas en Unity las cuales presentan un costo computacional elevado, principalmente en lo que refiere a visualización. Tanto en el proceso de implementación como en el de ejecución, se encontraron limitaciones en cuanto a la performance obtenida. Se procedió a reconocer los puntos críticos que incidían en la fluidez visual y se determinaron ciertos umbrales que no debían ser superados. Este análisis permitió modificar las aplicaciones desarrolladas y lograr un mejor funcionamiento en los distintos tipos de dispositivos móviles.

Por otra parte, la batería, componente indispensable en los dispositivos móviles, no ha sufrido grandes cambios en los últimos años. Los fabricantes han logrado

paulatinamente aumentar su capacidad y autonomía, pero sin avances notables. Actualmente se pueden conseguir celulares con baterías de hasta 5500 miliamperios hora (mAh) y autonomía de dos días. Dicha autonomía se estima a partir de la necesidad de suministro eléctrico requerido por los componentes internos del dispositivo, creando así un compromiso entre el poder de procesamiento, la visualización y la duración de la carga de la batería. Como consecuencia de esto, si una aplicación realiza un procesamiento intensivo, limitará el tiempo que el usuario puede usar el dispositivo sin necesitar una recarga.

Este mayor consumo plantea problemas para la evolución de la computación móvil, ya que los desarrolladores no pueden utilizar todo el potencial de la tecnología actual sin sacrificar la autonomía de la batería hasta una nueva recarga.

Por lo tanto, al momento de desarrollar una aplicación móvil, es de suma importancia tener en cuenta el consumo energético que pueda llegar a generar [3][4]. Esto toma mayor relevancia en las aplicaciones móviles 3D, dado que éstas son implícitamente demandantes de poder de cómputo y por consiguiente de consumo de energía.

A fin de ayudar a conseguir un equilibrio en las características relevantes de las aplicaciones móviles 3D y lograr una ejecución eficiente, se realizó el presente estudio donde los parámetros más representativos de los modelos 3D son puestos a prueba para medir su performance y consumo.

Características tales como el número de polígonos, la aplicación de luces y sombras, la utilización de texturas y/o transparencias, la visualización de sistemas de partículas y el cálculo de la física de objetos que componen la escena, son ejemplos de los principales ítems a evaluar. Se utilizarán entonces la misma serie de pruebas independientes definidas en [5] para evaluar la performance y consumo de energía.

La aplicación de este conjunto de pruebas simplifica considerablemente la tarea de determinar los puntos críticos a optimizar en las aplicaciones desarrolladas, posibilitando así la identificación del punto de equilibrio en la calibración de las características analizadas.

3 Propuesta de evaluación

El principal objetivo es determinar cuáles son las características de una aplicación 3D desarrollada en Unity y Unreal, que generan un mayor impacto en la performance de ejecución y en el consumo de energía en un dispositivo móvil.

Como se ha comentado previamente, las características de las aplicaciones móviles 3D que tienen relación directa con el procesamiento intensivo, también la tienen con el alto consumo de energía.

Con el fin de poder evaluar en forma aislada cada una de estas características en aplicaciones realizadas con el motor Unity, se desarrollaron dos prototipos móviles 3D.

El primer prototipo evalúa cómo las características mencionadas inciden en la performance, tiempo de respuesta y fluidez de ejecución de las aplicaciones desarrolladas con Unity. Esta tarea la realiza mediante un conjunto de pruebas en el que se testean cada una de las características de forma independiente [5]. A lo largo

de cada prueba, se va incrementando el número de objetos en la escena y se analiza el impacto generado.

El segundo prototipo evalúa cómo estas mismas características inciden en el consumo de energía de las aplicaciones desarrolladas con Unity. Esta tarea la realiza mediante una adaptación del conjunto de pruebas definido en [5] en el que también se testean cada una de las características de forma independiente [6].

La metodología de evaluación de consumo de energía implica relevar durante un período de tiempo prefijado el consumo de energía bajo la ejecución de cada una de las distintas pruebas. A diferencia de la metodología utilizada para la evaluación de performance, el número de objetos en la escena es constante durante el tiempo de ejecución de cada prueba. Una vez transcurrido el tiempo prefijado, se obtiene el valor del consumo generado.

Para hacer posible el análisis comparativo entre Unreal y Unity fue necesario implementar también ambos prototipos en el motor Unreal.

4 Desarrollo de prototipos

En lo que respecta a los prototipos en Unity, dichos prototipos fueron desarrollados previamente en los trabajos [5] y [6].

Unreal Engine es un framework de desarrollo de aplicaciones 3D que se destaca por su capacidad de manejar hasta un millón de partículas en una sola escena. Como lenguaje de programación utiliza C++ en combinación con un lenguaje visual denominado Blueprints.

La utilización de C++ en Unreal Engine brinda a los desarrolladores un gran control sobre las acciones de todo el sistema, pero a su vez vuelve el proceso de codificación mucho más complejo.

Los Blueprints, son un tipo de programación basada en nodos que permite a los desarrolladores implementar la lógica completa de una aplicación de una forma sencilla. Cada nodo tiene una determinada función con entradas y salidas, y al estar conectarlos entre sí se introduce la lógica necesaria. Los Blueprints pueden contener variables, tipos de datos, matrices, enumeraciones, mapas hash, funciones, macros, eventos personalizados, entre otros.

Utilizar C++ puro es más eficiente para aplicaciones de lógica compleja. Sin embargo, dada la sencillez de las pruebas del prototipo a desarrollar, no existe diferencia apreciable entre ambas alternativas de codificación. Por este motivo, los prototipos fueron desarrollados utilizando Blueprints.

En Unreal Engine existen tres perfiles básicos de calidad (alto, medio y bajo), y es posible configurar manualmente la calidad gráfica que tendrá cada perfil. Para cada familia del procesador y sistema operativo es posible elegir la calidad con la que debe ser ejecutada la aplicación. De todas formas, luego el motor de Unreal puede decidir ejecutar la aplicación en un perfil de calidad inferior si así lo considera necesario. Por esta razón, no es posible modificar la calidad gráfica en tiempo de ejecución, tal como se realiza en el prototipo desarrollado en Unity.

5 Pruebas realizadas y resultados obtenidos

Tanto para el estudio de performance como para el de consumo de energía la experimentación consistió en efectuar el conjunto de pruebas sobre una serie de dispositivos móviles con poder de cómputo diferente. Las pruebas se realizaron repetidas veces con cada prototipo y sobre cada dispositivo móvil, a fin de validar la consistencia de la medición.

El conjunto de pruebas es el definido en [5] y son las que se detallan brevemente a continuación:

- Prueba 1: renderización de objetos simples. Se presentan progresivamente en pantalla objetos simples sin textura en movimiento en una escena sin iluminación ni sombras. Los objetos deben rotar continuamente a velocidad constante.
- Prueba 2: renderización de objetos complejos. Consiste en visualizar un objeto complejo en movimiento, el cual debe contener un elevado número de polígonos. La distancia de renderizado (*clipping plane*) se va incrementando a medida que avanza la prueba.
- Prueba 3: luces y sombras. Se realiza una simulación similar a la renderización de objetos simples, pero en este caso la escena contiene iluminación y objetos con proyección y recepción de sombras.
- Prueba 4: texturas. Se realiza una simulación similar a la renderización de objetos simples, pero en este caso los objetos poseen texturas complejas, como puede ser transparencias, reflejos, etc.
- Prueba 5: sistemas de partículas. Se crea una escena en donde se presenta progresivamente nuevas instancias de un sistema de partículas (por ejemplo humo, chispas, explosión, etc.).
- Prueba 6: física. Se realiza una simulación similar a la renderización de objetos simples, pero en este caso a los objetos se les aplica reglas de física, como por ejemplo *gravedad*.

En lo que respecta a las pruebas relacionadas con performance, de cada prueba se almacenaron los valores de cuadros por segundo (FPS) logrados a medida que se incrementa la cantidad de objetos en la escena, hasta llegar a un máximo de 8192 objetos. En el caso particular de la prueba 2, en el que sólo se cuenta con un único objeto complejo, se almacenaron los FPS logrados a medida que se visualiza una mayor porción del objeto complejo. En la figura 1 se puede observar la prueba 6 ejecutándose en el nuevo prototipo desarrollado en Unreal Engine.

Las pruebas de consumo de energía se realizaron configurando el número de cuadros por segundo a un número fijo, dado que la cantidad de FPS utilizados tienen una incidencia directa sobre el consumo, tal como se verifica en [6]. A su vez, el número de objetos fijado para cada prueba se estableció con el fin de garantizar que todos los dispositivos usados en un posible futuro, incluso los de menor poder de procesamiento, puedan ejecutar fluidamente el conjunto de pruebas. Las pruebas en cuestión se realizaron en cada ocasión durante varios minutos, para minimizar potenciales errores en la toma de datos debido a eventuales desviaciones temporales. En la figura 2 se puede observar la prueba 1 ejecutándose en el nuevo prototipo desarrollado en Unreal Engine.

Como premisa básica, en todos los casos - y con la finalidad de reducir potenciales errores - se definieron las siguientes condiciones a lo largo de las ejecuciones:

- Al comenzar con el conjunto de pruebas, los dispositivos a utilizar deben tener la carga de la batería mayor al 95%.
- Las ejecuciones deben realizarse con el brillo de la pantalla del dispositivo al mínimo.
- Los dispositivos deben estar configurados en “Modo avión”, para evitar cualquier tipo de recepción de señal que pudiera incidir en las mediciones.
- No deben existir aplicaciones ejecutándose en segundo plano, dado que pueden consumir tiempo de procesamiento y de esa forma afectar las mediciones.

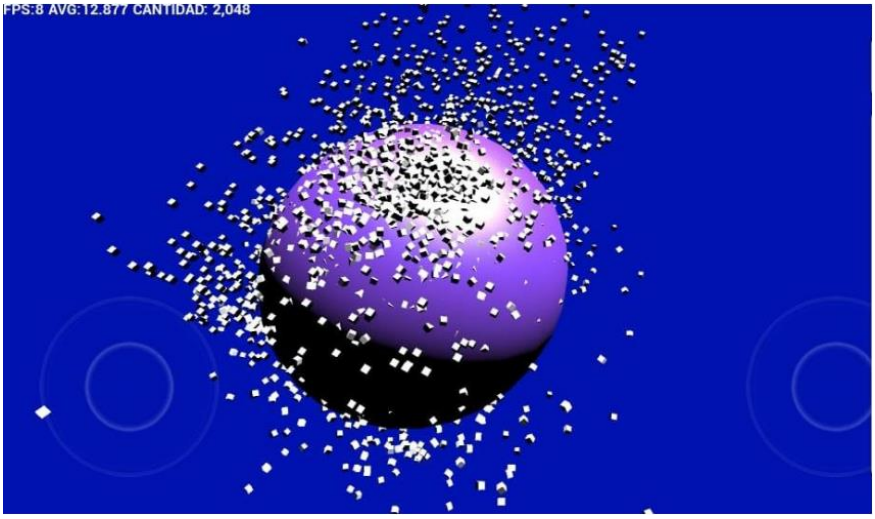

Fig. 1. Prototipo de performance en Unreal Engine. Prueba 6.

Un aspecto importante a tener en cuenta bajo Unreal - y que en cierta medida puede afectar tanto a performance como a consumo - es la falta de optimización en los proyectos orientados a dispositivos móviles, básicamente debido a la gestión de Draw Calls [7] que el motor realiza.

La herramienta de profiling seleccionada para realizar las mediciones de consumo de energía en [6] fue Trepn Profiler. Esta herramienta permitía medir en tiempo real el consumo de energía de una aplicación en particular, asegurando una medición precisa de la energía consumida en dispositivos que cuenten con procesadores desarrollados por Qualcomm Technologies Inc., empresa desarrolladora de la herramienta. En la actualidad, dicha empresa reemplazó la herramienta Trepn Profiler por una nueva herramienta denominada Snapdragon Profiler [8]. Sin embargo, utilizando esta herramienta no se obtuvieron resultados estables, por lo que se buscaron alternativas, y finalmente se optó por usar la aplicación AccuBattery. Esta aplicación, brinda información precisa sobre el consumo de batería en miliamperios hora (mAh).

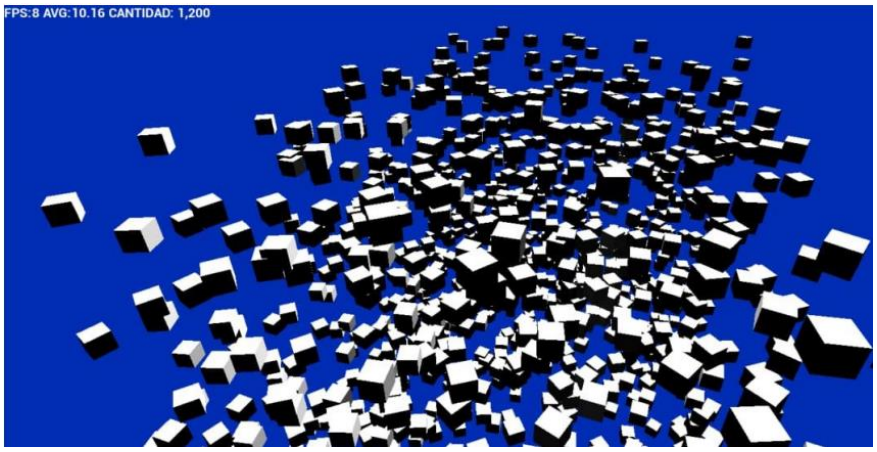

Fig. 2. Prototipo de consumo en Unreal Engine. Prueba 1.

Debido al cambio de la herramienta de medición de consumo, los resultados obtenidos en [6] no pudieron ser utilizados. Se realizaron nuevamente todas las pruebas en el prototipo desarrollado en Unity usando la herramienta de medición AccuBattery. Luego, se realizaron las pruebas en el nuevo prototipo desarrollado en Unreal también utilizando AccuBattery. De esta forma los resultados obtenidos son válidos para el análisis comparativo.

Las tablas 1 y 2 presentan los resultados de la degradación en la performance - medida en cuadros por segundo - en función del aumento de la complejidad computacional de cada una de las pruebas realizadas.

La figura 3 muestra el detalle de la degradación de la performance en Unity para el conjunto de casos de prueba definido, mientras que la figura 4 muestra este mismo detalle bajo Unreal Engine. La tabla 3 muestra los valores obtenidos en el estudio de consumo energético para el conjunto de pruebas realizado sobre ambos frameworks de desarrollo. La figura 5 presenta una comparativa de consumo energético sobre el conjunto de casos de prueba definido, abarcando los valores obtenidos tanto en Unity como en Unreal.

Tabla 1. Degradación de la performance en las pruebas realizadas en Unity.

Performance (FPS)	Unity												
Objetos simples	60	60	60	60	60	60	60	60	60	47	29	16	8	4
Objetos complejos	40	38	34	24	17	11	8	6	5	5	4	4	4	4
Luces y sombras	60	60	60	60	60	60	60	60	59	40	29	15	8	4
Texturas	60	60	60	60	60	60	60	60	60	47	26	13	8	4
Partículas	60	60	60	60	46	27	14	7	4	3	3	0	0	0
Físicas	60	60	60	60	60	60	60	60	60	45	8	3	3	3

Tabla 2. Degradación de la performance en las pruebas realizadas en Unreal Engine.

Performance (FPS)	Unreal Engine													
Objetos simples	60	60	61	60	60	60	58	50	40	28	18	10	8	8
Objetos complejos	8	8	8	8	8	8	8	8	8	8	8	8	8	8
Luces y sombras	60	60	60	59	60	60	59	48	42	30	20	10	8	8
Texturas	60	60	60	60	61	58	56	46	37	29	18	12	8	8
Partículas	19	19	19	18	16	14	11	9	8	8	8	8	8	8
Físicas	60	59	60	60	60	60	58	50	42	30	18	10	8	8

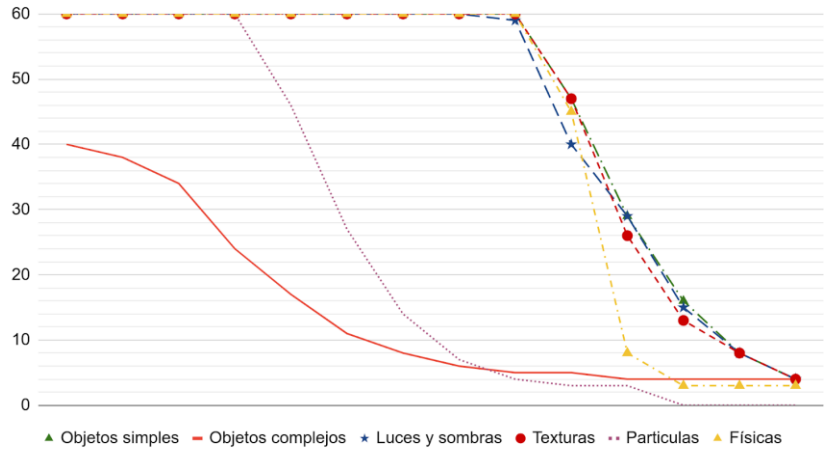

Fig. 3. Representación gráfica de la degradación de la performance en Unity para el conjunto de pruebas realizado.

Tabla 3. Consumo en las pruebas realizadas en Unity y Unreal Engine.

Consumo (mAh)	Unity	Unreal
Objetos simples	19.8	51.53
Objetos complejos	27.18	61.06
Luces y sombras	20.73	37.4
Texturas	20.23	47.48
Partículas	15.94	21.78
Físicas	18.36	18.65

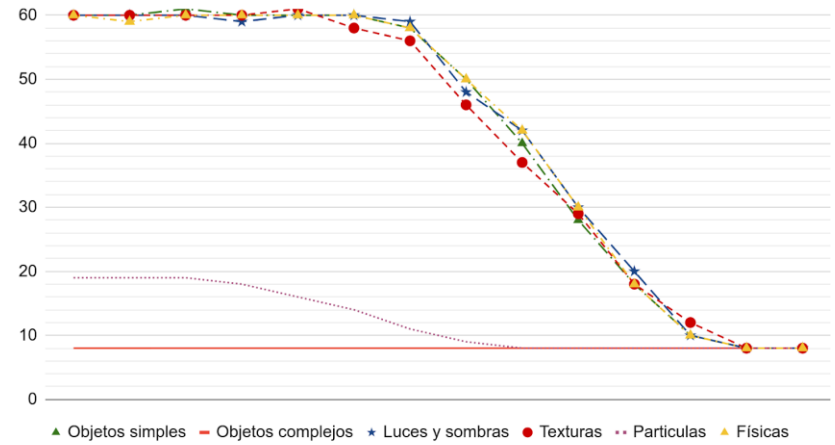

Fig. 4. Representación gráfica de la degradación de la performance en Unreal para el conjunto de pruebas realizado.

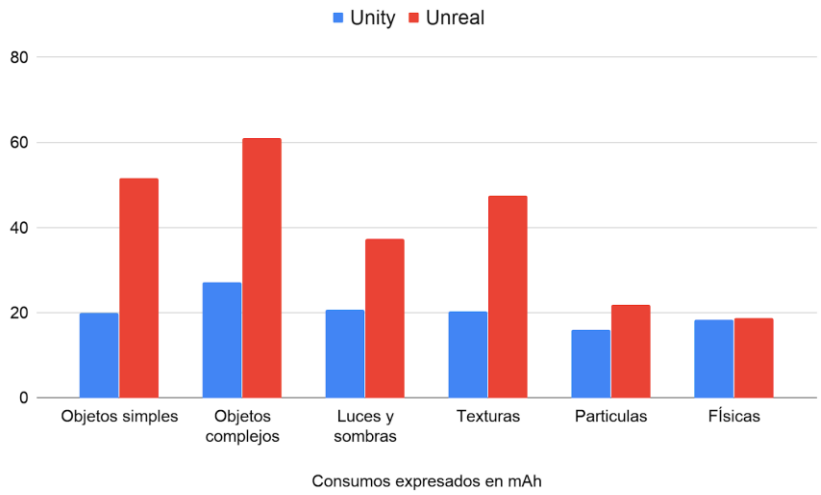

Fig. 5. Consumo Unity vs. Unreal Engine.

6 Conclusiones y trabajo a futuro

El presente trabajo presenta un conjunto de actividades para evaluar y comparar la performance y el consumo de energía de aplicaciones 3D sobre dispositivos móviles, específicamente para los frameworks Unreal Engine y Unity.

La evaluación propuesta permite determinar la incidencia de cada una de las características principales de las aplicaciones 3D sobre la fluidez de visualización y el consumo energético.

De esta manera, un ingeniero de software puede desarrollar estas aplicaciones teniendo en cuenta los considerandos aquí detallados, tales como cantidad de polígonos, texturas, sombras, entre otras características visuales.

Se desarrollaron dos prototipos en el framework Unreal Engine, que permiten ejecutar una serie de pruebas independientes para cada una de las características típicas de una aplicación 3D. Estos resultados fueron posteriormente contrastados con los obtenidos en las aplicaciones análogas desarrolladas previamente en el framework Unity.

Se determinó que - en términos generales - Unreal Engine tiene la capacidad de desarrollar y procesar gráficos más potentes y realistas que Unity, pero generando un mayor requerimiento de recursos y energía.

A futuro se prevé ampliar esta comparativa a otros frameworks de este tipo, como por ejemplo CryEngine [9], el cual, si bien actualmente no tiene la popularidad de Unity o Unreal, se encuentra en constante crecimiento.

Referencias

1. Unity. https://unity3d.com.
2. Unreal Engine. https://www.unrealengine.com.
3. Riaz, M. "Energy consumption in hand-held mobile communication devices: A comparative study". International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). 2018. ISBN: 978-1-5386-1370-2/18.
4. Corbalan L., Fernández J., Cuitiño A., Delia L., Cáseres G., Thomas, P.; Pesado, P. "Development Frameworks for Mobile Devices: A Comparative Study about Energy Consumption". Association for Computing Machinery (ACM). 2018. ISBN 978-1-4503-5712-8/18/05.
5. Cristina, F.; Dapoto, S.; Thomas, P.; Pesado, P. “Performance evaluation of a 3D engine for mobile devices”. Computer Science – CACIC 2017. Communications in Computer and Information Science, vol 790. A. De Giusti, Springer International Publishing. 2018. ISBN: 978-3-319-75213-6, 978-3-319-75214-3, pages 155-163.
6. Cristina, F.; Dapoto, S.; Thomas, P.; Pesado, P. "Análisis de consumo de energía en aplicaciones 3D sobre dispositivos móviles". XXIV Congreso Argentino de Ciencias de la Computación (CACIC). 2018. ISBN: 978-950-658-472-6, páginas 622-630.
7. Unreal Engine. CPU Profiling. https://docs.unrealengine.com/en-US/Engine/Performance/CPU/index.html.
8. Snapdragon Profiler. https://developer.qualcomm.com/software/snapdragon-profiler.
9. CryEngine. https://www.cryengine.com.