UNIVERSIDADE DA CORUÑA

PROGRAMACIÓN DE SISTEMAS 22/23 Q2

FloatAR

Autores:
Antonio Vila Leis - antonio.vila@udc.es
Hugo Sanjiao Varela - h.sanjiao@udc.es
Breogán Fernández Moreira - breogan.fernandez@udc.es
Óscar Alejandro Manteiga Seoane - oscar.manteiga@udc.es

Fecha: *A Coruña, 12 Mayo 2023*

Índice

Capítulos **Página**

Cuadro 1: Tabla de versiones.

Versión	Fecha	Autor
1	16 de marzo de 2023	Antonio — Hugo — Breogán — Óscar
2	18 de abril de 2023	Antonio — Hugo — Breogán — Óscar
3	9 de mayo de 2023	Antonio — Hugo — Breogán — Óscar
4	12 de mayo de 2023	Antonio — Hugo — Breogán — Óscar

1. Introducción

La aplicación FloatAR es una aplicación de realidad aumentada del juego "Hundir la flota"para dispositivos Android. Esta aplicación utiliza la API ARCore y se desarrollará para un trabajo de clase de la asignatura Programación de Sistemas en un grupo de 4 personas. El juego podrá ser utilizado de forma estándar o con el uso del modo de realidad aumentada.

1.1. Objetivos

El objetivo principal de FloatAR es proporcionar una experiencia de juego más inmersiva para los usuarios utilizando la tecnología de realidad aumentada. Esta tecnología no está muy expandida en las aplicaciones de Android, pero aun menos en los videojuegos, por lo que otro objetivo implícito podría ser la investigación de esta tecnología. Los objetivos secundarios incluyen:

- Desarrollar de forma convencional el juego de "Hundir la flota.[en] modo local.
- Implementar el modo para 2 jugadores vía Internet.
- Integrar un sistema de puntuación y registro de puntuación para los usuarios.
- Añadir el espacio de juego al mundo real con la realidad aumentada utilizando la API ARCore [1].

1.2. Motivación

La tecnología de realidad aumentada ha ganado popularidad en los últimos años y se está utilizando cada vez más en juegos y aplicaciones. El uso de la realidad aumentada en juegos como hundir la flota puede proporcionar una experiencia de juego más inmersiva y emocionante para los usuarios. A mayores, la investigación e implementación de esta tecnología en juegos es un apartado que nos produce gran interés, ya que no está muy extendida y creemos que será muy usada en el futuro. La aplicación FloatAR también puede ser aplicable en otros juegos que requieren que ambos jugadores (o más de 2) no puedan ver lo que ven los demás. Esto hace que en un futuro próximo podamos expandir el juego si es necesario.

1.3. Trabajo relacionado

Hay otras aplicaciones de realidad aumentada para juegos, como Pokémon Go (Niantic) [2], la aplicación más exitosa que implementa realidad aumentada (y una de las más exitosas a nivel de aplicación general) que utilizan esta tecnología para crear una experiencia de juego más inmersiva. Otros ejemplos son Angry Birds AR: Isle of Pigs (Rovio Enterteinment) [3], Jurassic World Alive (Ludia Inc.) [4], Harry Potter: Wizards Unite (Niantic) [5], Minecraft Earth (Mojang) [6]... Todas ellas implementan la RA, pero de formas muy distintas:

- Pokémon GO, Jurassic World Alive y Harry Potter: Wizards Unite la usan para mostrar sus Pokémons, dinosaurios o entidades fantásticas.
- Minecraft Earth la usa para mostrar su mundo en nuestra habitaciones o espacios de la vida real a través de la cámara.
- Angry Birds AR: Isle of Pigs y nuestra aplicación FloatAR la usan para mostrar el espacio de juego en la vida real para ver como se desarrolla el evento.

2. Análisis de requisitos

Antes de comenzar el desarrollo de la aplicación FloatAR, se debe realizar un análisis preliminar de los requisitos. Esto ayudará a definir las funcionalidades clave de la aplicación y establecer prioridades.

- Jugar de forma local contra la IA sin realidad aumentada.
- Jugar contra otro jugador sin realidad aumentada.
- Interfaz de usuario intuitiva y simple con sistema de puntuación.
- Modo de realidad aumentada.

2.1. Funcionalidades

1. Tablero de juego sin ser en realidad aumentada.
2. Colocación de barcos en el tablero con sistema de colisiones.
3. Menú de ajustes, de información y de ayuda.
4. Juego local contra la IA, con turnos, detección de victoria/derrota...
5. Juego online contra otro jugador con las mismas características.
6. Posibilidad de jugar a algún modo en realidad aumentada.

2.2. Prioridades

1. Crear un tablero sin ser en realidad aumentada (Core).
    a) Permitir al usuario colocar sus barcos en el tablero virtual (Core).
    b) Implementar un sistema de detección de colisiones para asegurar que los barcos no se superpongan (Core).
2. Crear pantalla de ajustes, ayuda e información sobre la aplicación (Core).
3. Implementar sistema de juego local (Core).
    a) Implementar un sistema de juego por turnos contra una IA (Core).
    b) Implementar un sistema de detección de victoria y derrotas (Core).
4. Implementar sistema de juego online (Core).
    a) Implementar el sistema de comunicación entre jugadores para sincronizar el juego y las puntuaciones/eventos (Core).
    b) Implementar un sistema de juego por turnos contra otro jugador (Core).
5. Desarrollar la realidad aumentada (Core).
    a) Desarrollar la parte de realidad aumentada.
6. Mostrar la puntuación y el estado del juego en tiempo real (Accesorio).
7. Implementar un sistema de animación para ciertas acciones (Accesorio).

3. Planificación inicial

En esta sección se establecerá una planificación muy simple y básica del proyecto, definiendo las iteraciones, responsabilidades, hitos y entregables.

3.1. Iteraciones

Consideramos 12 semanas para la implementación de las funcionalidades del juego:

- Inicio del proyecto, organización, reparto de trabajo y funcionalidades 1, 2 y 3 (base de aplicación y juego). [4 semanas].
- Funcionalidades 4 y 5 (sistema de juego offline/online) [4 semanas].
- Funcionalidad 6 (realidad aumentada) [4 semanas].

3.2. Responsabilidades

- Óscar Alejandro Manteiga Seoane:
  1. Desarrollo de la base de la aplicación (pantallas, ajustes...).
  2. Tablero básico de juego.
  3. Prueba y estudio de ARCore para implementarla una vez llegada su implementación.
- Antonio Vila Leis:
  1. Desarrollo del sistema de colocación de barcos en el tablero.
  2. Desarrollo del sistema de juego contra la IA.
  3. Prueba y estudio de ARCore para implementarla una vez llegada su implementación.
- Hugo Sanjiao Varela:
  1. Desarrollo de la base de la aplicación (pantallas, ajustes...).
  2. Desarrollo del sistema de juego multijugador.
  3. Redacción de la documentación necesaria.
- Breogán Fernández Moreira:
  1. Desarrollo del sistema de juego contra la IA.
  2. Desarrollo del sistema de juego multijugador.
  3. Desarrollo de los test.

Es importante mencionar que la implementación de la realidad aumentada la intentaremos entre todos ya que ninguno tiene experiencia desarrollándola. Serán Óscar y Antonio los encargados de su estudio a lo largo del desarrollo para poder partir de una base de conocimiento sobre la API ARCore. Las responsabilidades de cada uno pueden variar a lo largo del proyecto.

3.3. Hitos

1. Entrega de la primera versión de la memoria, con la planificación inicial y requisitos/funcionalidades.
2. Entrega de la base de la aplicación y del juego.
3. Entrega del juego completo sin realidad aumentada.
4. Entrega del juego con la realidad aumentada.

En este tercera entrega, no hemos podido completar el hito marcado, ya que el modo multijugador no funciona completamente. Como le falta podo desarrollo, intentaremos entregarlo junto con la Realidad Aumentada en la última entrega. En caso de que no nos de tiempo, lo que haremos en lugar de poder jugar en realidad aumentada, será hacer una pequeña demo de su uso.

3.4. Incidencias

Posibles incidencias y planes de contingencia: En caso de problemas genéricos durante el desarrollo, se realizarán pruebas (tests) durante la implementación para intentar corregirlos antes de cada entrega. En caso de encontrarlos, se buscarán alternativas y soluciones en conjunto con todo el equipo. El desarrollo del juego básico lo sabemos hacer, por los que los principales inconvenientes se pueden generar con la implementación de la realidad aumentada, la sincronización online de jugadores y la sincronización de la realidad aumentada.

En el caso de la implementación de la realidad aumentada en el proyecto, si no conseguimos un correcto funcionamiento de esta tecnología, se pasará a desarrollar una aplicación de juegos de mesa online. Los juegos que incorporemos serán varios y distintos entre si para incrementar la complejidad del proyecto.

En caso de poder implementar la realidad aumentada pero sin que sea de forma sincronizada por internet, se creará el juego de hundir la flota de forma local para visualizar tu parte del tablero en esta tecnología mientras que de fondo y para sincronizar el online se juega con el modo básico sin RA.

En el caso de la sincronización online de jugadores, es posible que se produzcan problemas de conexión que afecten a la experiencia de juego de los usuarios. Para minimizar este riesgo, se planearán y realizarán pruebas rigurosas antes de lanzar la aplicación para asegurar la estabilidad del servidor y la capacidad de manejar múltiples conexiones simultáneas.

En cuanto a la sincronización de la realidad aumentada, también pueden surgir problemas técnicos que afecten a la experiencia de los usuarios. Si estos problemas son significativos, se puede optar por desactivar la funcionalidad de realidad aumentada y ofrecer a los usuarios una experiencia de juego sin esta característica.

Otro posible problema es la incompatibilidad de la aplicación con ciertos dispositivos móviles. Para evitar esto, se investigará y se tendrán en cuenta las especificaciones y requisitos de hardware de los dispositivos móviles más utilizados y se realizarán pruebas de compatibilidad antes del lanzamiento.

4. Diseño

4.1. Elementos utilizados

- Botones: obviamente, este elemento tiene que estar en cualquier aplicación donde el usuario pueda interaccionar con la misma. Los usamos a lo largo de todas las vistas, pero donde más relevancia tienen son en los tableros. Cuando

desarrollamos el código para poder jugar en modo local teníamos los tableros rellenados con un botón por cada casilla en el .xml correspondiente. Esto nos daba muchos problemas de rendimiento e intentamos migrar de lo botones a otros elementos sin éxito. Al final, la solución que usamos es la generación de esos botones de los tableros en el .java.

- Imagen: la utilizamos para mostrar el logo de la aplicación tanto en la actividad principal como en la de información de la aplicación.
- Switch: para activar o desactivar las notificaciones y activar o desactivar el modo oscuro utilizamos 2 switches en la actividad de los ajustos. Decidimos que era la mejor idea ya que es un elemento al que el usuario está familizarizado y facilita la implementación.
- SeekBar: para controlar el volumen usamos un slider. Es la mejor opción ya que el propio Android lo usa también.
- Spinner: en la última entrega incorporaremos traducciones de la aplicación, por lo que el uso de este spinner faciltará la elección de idioma que quiera el usuario.
- LinearLayout: usaremos esta disposición en la mayoría de actividades de la aplicación por su sencillez y facilidad de organizar los elementos en la pantalla.
- RelativeLayout: lo usamos en la actividad de información de la aplicación por su típico uso en estos casos. De esta forma queda todo centrado en la pantalla sin dificultad ninguna.
- ScrollView: en la actividad de ayuda usamos una scroll view para que el usuario pueda leer todo el texto haciendo un simple e intuitivo scroll de la pantalla.
- TextView: elemento necesario para mostrar texto.
- EditText: elemento necesario para que el usuario pueda introducir texto.

4.2. Arquitectura

La arquitectura de la aplicación se basará en la utilización de diferentes componentes que trabajarán en conjunto para ofrecer una experiencia completa al usuario. Estos componentes serán los siguientes:

- Actividades: la aplicación contará con varias actividades que permitirán al usuario acceder a diferentes secciones de la aplicación. Por ahora tenemos las siguientes:

    1. Activity de inicio: se ejecutará cuando se inicie la aplicación y mostrará la pantalla de inicio con opciones para iniciar un nuevo juego, cargar un juego guardado (sin confirmar esta funcionalidad) o ver las opciones de configuración.

2. Activity de creación del tablero: mostrará una pantalla donde el usuario podra colocar sus barcos sobre su tablero, en esta actividad también se crea el tablero del rival para el modo de un jugador.
3. Activity de juego "singleplayer": mostrará el tablero del jugador y permitirá jugar contra la computadora.
4. Activity de juego "multiplayer": mostrará una lista de las puntuaciones más altas de los jugadores.
5. Activity de configuración: permitirá a los usuarios personalizar la configuración de la aplicación, como cambiar el idioma, el nivel de dificultad (sin confirmar esta funcionalidad), la configuración de sonido, etc.
6. Activity de ayuda: mostrará información de ayuda y tutorial para los nuevos usuarios.
7. Activity de información: proporcionará información de la aplicación y de los autores de la misma.
8. Activity de lobby: mostrará una serie de salas a las que el usuario puede unirse para conectarse a un juego multijugador, asi como la posibilidad de crear una nueva sala.
9. Activity de fin del juego: mostrará el resultado de la partida (si se ha ganado o perdido) y un botón para regresar a la activity de inicio.

- Servicios: para poder jugar online, usamos Firebase Realtime Database como servicio para sincronizar los arrays de los tableros de ambos jugadores, las salas de juego y el almacenamiento de datos que puedan ser útiles en un futuro (nombre de jugador, resultado de la partida...). Para acceder a esto, fue necesario registrarse en Firebase, crear una aplicación en su consola, descargar unos ajustes para poder funcionar con la aplicación de Android Studio y crear una Realtime Database donde guardar los datos.

- Broadcast receivers: se utilizarán broadcast receivers para gestionar las notificaciones push que se envíen al usuario.

4.3. Persistencia

Como mencionamos en el punto anterior, la persistencia de datos online la hacemos con Firebase Realtime Database. En cuanto a los datos en local (partidas contra la máquina, preferencias...) usamos el almacenamineto del dispositivo de cada usuario.

4.4. Vista

La aplicación contará con varias actividades y fragmentos que permitirán al usuario acceder a diferentes secciones de la aplicación. Además, se incluirán notificaciones

en el último entregable. A continuación, se describen los elementos visuales que se incluirán en la aplicación:

- Actividades: la aplicación contará con varias actividades, como la creación del tablero, la ejecución de las partidas, el menú principal y las pantallas de ayuda o configuración.
- Diálogos: se utilizaran diálogos para que el usuario pueda ingresar el nombre de una nueva sala que se disponga a crear o para unirse a una existente y poder ingresar su nombre.
- Menús: casi la totalidad de las actividades tendrán un menú en la parte superior para acceder a los ajustes, ayuda o información sobre la aplicación. A mayores, si no estamos en la actividad principal, también tendremos una flecha para retroceder a la actividad anterior.

4.5. Comunicaciones

Las comunicaciones con el servidor de Firebase se hará a través del enlace que nos proporciona esta herramienta para poder acceder a la base de datos. Queda pendiente para la última entrega realizar una autenticación de usuarios e incremento de la seguridad, ya que actualmente la base de datos está abierta a posibles ataques y borrado de información.

4.6. Sensores

Los sensores utilizados serán gestionados con la API ARCore [1], aunque no sabemos por ahora cuales son, ya que su implementación se realizará en el último entregable.

4.7. Trabajo en background

Por ahora la aplicación no cuenta con trabajo en segundo plano, aunque se planea añadirlos para gestión de notificaciones en la última entrega.

Referencias

[1] "Arcore sdk for android," 2021.

[2] "Pokemon go," 2016.

[3] "Angry birds ar: Isle of pigs," 2018.

[4] "Jurassic world alive," 2018.

[5] "Harry potter: Wizards unite," 2018.

[6] "Minecraft earth," 2018.