Gravity DS

Proyecto de Estructura de Computadores

Facultad de Informática (EHU/UPV)

Curso 2011-2012

Proyecto desarrolado por Puma Corporation™, cuyos integrantes son:

Daniel Franco Barranco	Asier Mujika Aramendia	Iván Matellanes Pastoriza
E-mail: dfranco007@ikasle.ehu.es	E-mail: amujika009@ikasle.ehu.es	E-mail: imatellanes003@ikasle.ehu.es

Profesora: **Edurne Larraza Mendiluze**

Donostia, a 16 de Mayo de 2012

Resumen

Gravity DS es un videojuego de habilidad para Nintendo DS cuyo principal objetivo es evitar caer de las plataformas y aguantar el mayor tiempo posible jugando. El jugador posee el control del sentido gravitatorio, siendo ésta la única herramienta de la que dispone para mantenerse con vida. No se puede controlar el movimiento del personaje, es automático. Lo fundamental es evitar los obstáculos que se encuentran por el camino. Es necesario poner los cinco sentidos en la pantalla, y contar con destreza y reflejos para lograr una gran cantidad de puntos y entrar en el *ranking* de mejores puntuaciones. Opcionalmente, las monedas que aparecen de forma aleatoria a lo largo de la pantalla se pueden coger pulsando la pantalla táctil y suponen una bonificación de puntuación. El entretenimiento está garantizado.

Palabras clave: videojuego, gravedad, plataformas, aguantar

1. Introducción

Este documento explica la creación del proyecto "Gravity DS" en la asignatura Estructura de Computadores. La metodología utilizada para llevar a cabo este proyecto se ha basado en la teoría impartida en las clases de la asignatura, teoría de Entrada/Salida, DMA y Buses.

Para realizar este proyecto hemos hecho uso de todo lo aprendido en la parte de Entrada/Salida de la asignatura Estructura de Computadores y no solo eso, también hemos aplicado conceptos de otras asignaturas, como por ejemplo, matrices de transformación (de Álgebra) para realizar las transformaciones que nos permiten hacer nuestras características animaciones.

En esta memoria primeramente se aclaran los objetivos del proyecto. Posteriormente, se analiza cómo se está estructurado el proyecto, explicando cada estado y viendo en qué se basan dichos estados. Tras esto, se expone cómo fue el proceso de desarrollo del juego. En la siguiente sección hablamos de las conclusiones a las que hemos llegado después de hacer el proyecto, respecto a nuestra visión personal y respecto al trabajo de grupo. Por último, se muestran como referencia bibliográfica todas las páginas web utilizadas para una mejora del proyecto y como fuente de información.

2. Objetivos

Los principales objetivos de este proyecto son los siguientes:

- Aprender cómo funciona el subsistema de entrada/salida en un computador real.
- Comprender y utilizar el entorno de trabajo de la Nintendo DS.
- Conocer de primera mano las etapas de desarrollo de un proyecto científico-técnico.
- Mejorar las capacidades de comunicación y trabajo en equipo.
- Perfeccionar las habilidades de expresión oral y escrita.
- Asentar las bases adquiridas de programación estructurada.

3. Estructura del proyecto

3.1 *Máquina de estados*

Para realizar este proyecto se usó el autómata que en un principio estaba definido para la aclaración de los estados que debería tener el juego. Una vez realizado lo básico del juego se decidió ir implementando nuevos estados y pequeños cambios para mejorar el juego. Por lo tanto se tuvo que rehacer el autómata para añadir estos últimos cambios. Este fue el resultado final (Figura **3.1**).

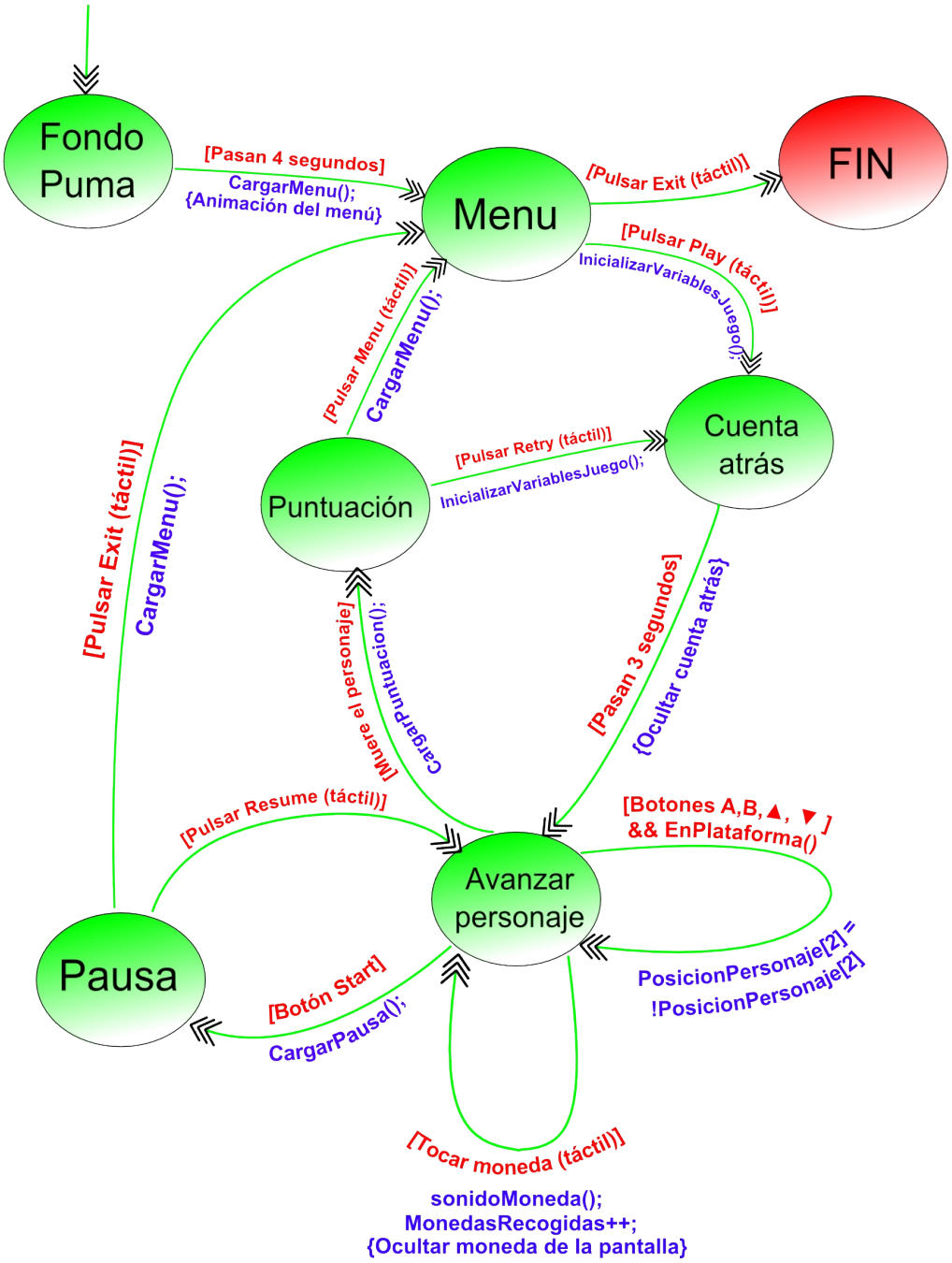

Figura 3.1. *Autómata del proyecto.*

3.2 *Análisis de los estados*

En este apartado se detalla el proyecto paso a paso y viendo cada estado. En cada uno se explica qué es lo que aparece en la pantalla y qué es lo que se debe hacer en cada caso.

3.2.1 *Fondo Puma*

En este primer estado, FONDO PUMA, se muestra en la pantalla principal el logotipo de nuestra corporación *PUMA CORPORATION* ™ y en la pantalla secundaria aparecerá la portada del juego. Este estado permanecerá hasta que pasados 3 segundos, el

logotipo se desvanecerá y se avanza al menú (Figura **3.2.1**).

3.2.2 *Menú*

En este segundo estado nos aparece en la pantalla principal dos botones: PLAY y EXIT. Como sus propios nombres indican el botón de EXIT nos llevará al estado FIN provocando el cierre del juego y el botón PLAY nos conducirá al estado CUENTA ATRÁS. Estas dos opciones se deberán escoger mediante la pantalla táctil.

Por otro lado en la pantalla secundaria tendremos unas pequeñas instrucciones del juego (Figura **3.2.2**).

Figura 3.2.1. *Estado FONDO PUMA.*

Figura 3.2.2. *Estado MENÚ.*

3.2.3 *Cuenta atrás*

En este tercer estado se mostrará un contador antes de empezar la pantalla. Para ello se mostrará una secuencia de números antes de empezar a jugar: “3, 2, 1, GO!” (Figura **3.2.3**). A continuación, se pasa al estado de AVANZAR PERSONAJE, aunque la partida realmente comienza con el GO!.

Por otro lado, en la pantalla secundaria aparecen las 10 mejores puntuaciones del juego, que se cargan desde el fichero *gravityds-scores.txt* situado en la carpeta *NDS* del sistema de archivos de la consola.

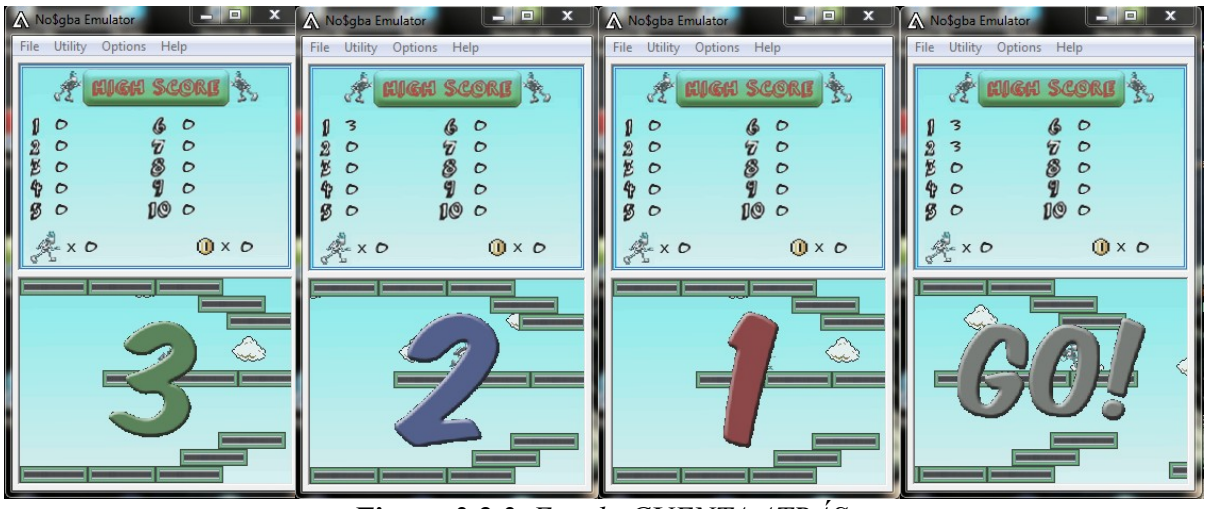

Figura 3.2.3. *Estado CUENTA ATRÁS.*

3.2.4 *Avanzar Personaje*

Este estado es en el que transcurre la mayor parte del juego. En la pantalla de arriba como antes se muestran las 10 mejores puntuaciones pero en este estado con un ligero cambio, los dos iconos de abajo donde aparecen el jugador y una moneda se van actualizando constantemente para saber en todo momento la puntuación que se está consiguiendo.

En la pantalla principal se mostrará la partida que consiste en sobrevivir el mayor tiempo posible cambiando la gravedad, utilizando las teclas A, B, Flecha Arriba o Flecha Abajo para esquivar los obstáculos, y además, por cada moneda cogida con la pantalla táctil, se añadirán puntos extra a la puntuación total del jugador (Figura **3.2.4**).

Cuando el jugador se quede rezagado o caiga al vacío se avanza al estado PUNTUACION.

Figura 3.2.4. *Estado AVANZAR PERSONAJE.*

3.2.5 *Pausa*

Este estado es una opción dentro del estado de AVANZAR PERSONAJE. Si durante la partida se desea parar el juego, pulsando el botón START del teclado de la Nintendo DS se accede a este estado llamado PAUSA en el que se parará el juego y saldrán dos botones en la pantalla principal: el botón RESUME que reanudará el juego donde lo habíamos dejado y el botón EXIT que conducirá al estado MENU de vuelta.

3.2.6 *Puntuación*

Este último estado informa en la pantalla principal del número de monedas recogidas durante la partida y la distancia recorrida. Con estos dos datos se calculará la puntuación total que el jugador ha obtenido; este último estado también brinda la oportunidad de volver a jugar otra vez pulsando el botón RETRY mediante la pantalla táctil que conduce al estado CUENTA ATRAS, y de volver al MENU principal del juego pulsando el botón MENU.

Por otro lado la pantalla secundaria se actualizará poniendo los puntos totales obtenidos en su respectivo lugar.

4. Desarrollo del juego

El desarrollo del juego comenzó a partir de la plantilla de proyecto que se encuentra en Moodle. Con el objetivo de hacer el código más legible, se acordó crear nuevos archivos adicionales a los de la plantilla para cada estado definido en el autómata (Figura **3.1**), donde cada archivo se encargase de controlar las acciones de cada estado y sus transiciones.

El primer paso fue abrir un repositorio en la página web GitHub.com, que permite alojar proyectos *open source* de forma gratuita accesibles a través del protocolo Git. De esta forma se consigue una mejor coordinación en el equipo y más comodidad al trabajar todos sobre el mismo código. El proyecto se ha desarrollado utilizando el IDE Eclipse con la herramienta devKitPro.

4.1 ***Entrada/Salida***

La implementación del videojuego comenzó con la entrada/salida, que se divide en los siguientes periféricos:

4.1.1 *Pantalla táctil*

La gestión de este periférico se hace por encuesta. En el juego, interesa conocer el botón que se ha pulsado en un menú, y si se ha pulsado una moneda en una partida. Con este fin se definen tres funciones de encuesta de pantalla en el fichero *pantalla.c*, donde cada una devuelve un identificador para conocer el elemento pulsado (en caso de que se haya pulsado algo). Además, se evita por medio de una variable de control que se lea más de una vez una pulsación prolongada.

4.1.2 *Teclado*

Pese a que en un principio se decidieran gestionar las teclas de la consola por interrupción, se observó al testear el juego en la propia máquina que las teclas son demasiado sensibles y en ocasiones se producen múltiples interrupciones en una única pulsación de tecla. En virtud de este contratiempo, se decidió por unanimidad cambiar el sistema de gestión por completo y realizar el control de las teclas por encuesta.

Así pues, se recapacitó cuál sería la mejor forma de controlar el teclado, llegando a la conclusión de que una función en el fichero *estado_avanzar.c* sería suficiente, ya que es el único estado que hace uso del teclado como se muestra en la máquina de estados (Figura **3.1**).

Esta función toma las medidas necesarias cuando se pulsa una tecla, véase cambiar la gravedad o pasar al estado de PAUSA.

4.1.3 *Temporizadores*

Los temporizadores han sido útiles para controlar el paso de los segundos en el momento de hacer la cuenta atrás y de hacer las animaciones. Además, se ha utilizado un segundo temporizador para los efectos sonoros, que cambian de volumen y/o frecuencia en un corto intervalo de tiempo, menor que un segundo.
Para su configuración y uso, se han implementado en *temporizadores.c* varias funciones de control: iniciar, detener, establecer frecuencia y gestión de interrupciones.

4.1.4 *Gestión de interrupciones*

Las rutinas encargadas de atender las interrupciones de temporizadores y las interrupciones Vblank se establecen en el fichero *interrupciones.c* mediante los mecanismos que la librería *libnds* proporciona para ello.

4.2 ***Gráficos***

Uno de los puntos fuertes del desarrollo de la aplicación han sido los gráficos.
Por una parte, se han usado sprites para la mayoría de los objetos móviles que aparecen en el juego. Para facilitar la tarea de creación de sprites, uno de los componentes del grupo, experimentado en el lenguaje Python, desarrolló un script que convierte una imagen en formato PNG en una secuencia de bits y su paleta de colores, en el formato adecuado para trabajar en la Nintendo DS. Dado que se utiliza una paleta diferente para cada sprite, se tomó la decisión de usar un banco de memoria virtual de 16 KiB para almacenar las paletas de los sprites. Así pues, hubo que modificar el mapeo inicial de los bancos de memoria para ajustarlos a las necesidades de los sprites.
Por otro lado, algunos objetos móviles, y sobre todo los estáticos, se implementaron como fondos en el fichero *fondos.c*. La cuenta atrás está hecha a partir de fondos pues facilitan el ajuste de su tamaño y posición para crear la animación. Para copiar estos fondos a la memoria se hace uso de la interfaz DMA, mucho más rápida que una copia regular.
Para coordinar ambos métodos de mostrar gráficos y evitar *glitches*, se limitan las ejecuciones del bucle principal del juego a una por fotograma. Esto es posible gracias a las interrupciones Vblank. La máquina envía una de estas interrupciones cada vez que se actualiza la pantalla, y el juego espera a que esto ocurra antes de seguir procesando el siguiente fotograma y llamar a las funciones de actualización de sprites y fondos que proporciona *libnds*.

4.3 S*onido*

Los elementos de sonido en este juego son bastante sencillos. Se reproducen un par de efectos sonoros: uno durante la CUENTA ATRÁS, que aumenta de frecuencia en el último segundo, al mostrar GO!; y otro cuando el jugador pulsa sobre una moneda en el transcurso de la partida.
El primer sonido va decreciendo en volumen por cada fotograma hasta que su volumen es nulo. El segundo sonido, el de la moneda, se ajusta por medio del temporizador número 1, que interrumpe 15 veces por segundo y además de decrecer en volumen, modifica la frecuencia para crear un sonido más agudo al estilo del efecto sonoro del clásico Mario Bros.

4.4 ***Lógica del juego***

La lógica del juego aunque a simple vista puede parecer compleja y enrevesada, no lo es tanto.
En el estado AVANZAR PERSONAJE se aumenta constantemente una variable global de distancia recorrida, que también sirve como puntuación. En base a esta distancia, se calcula cuáles son las plataformas que se deben mostrar en pantalla y en qué posición deben dibujarse, a partir de una lista de plataformas que componen el nivel del juego y que ha sido diseñada utilizando la herramienta *Blender*.

Otra variable global definida también en *estado_avanzar.c* indica la posición del personaje en la pantalla y el sentido en el que se encuentra la gravedad. Gracias a esta variable se dibuja el sprite de personaje en la pantalla, y un par de funciones auxiliares detectan posibles colisiones del personaje con las plataformas, modificando en caso de que las hubiera la posición del personaje y dando esa impresión de que el personaje se "bloquea" al chocar o se "apoya" en una plataforma. Esta variable también sirve para comprobar si el personaje ha muerto y pasar al estado PUNTUACION. El sentido de la gravedad se cambia modificando la variable correspondiente entre los valores 0 y 1, y sólo se puede hacer mientras se está apoyado en una plataforma.
Las nubes de fondo y las monedas aparecen de forma pseudoaleatoria por la pantalla, siempre dentro de unos márgenes para no desbordar al jugador. Los números pseudoaleatorios se consiguen con la función *rand()* de la librería estándar de C.
La puntuación se calcula en base a la distancia recorrida y al número de monedas recogidas, siendo la fórmula utilizada:

*Puntuación = (DistanciaRecorrida / 100) + (MonedasRecogidas * 2)*

4.5 *Sistema de archivos*

Por último, con el fin de guardar las mejores puntuaciones obtenidas durante la partida y que no se perdieran al apagar la consola, se implementó un método para acceder al sistema de archivos de la NDS con ayuda de la libreía *libfat*, parte de devKitPro.
Al iniciar el juego se procede a leer los datos guardados en el fichero de texto */NDS/gravityds-scores-txt* y cada vez que se termina una partida se escriben en el fichero las 10 mejores puntuaciones hasta el momento, pudiendo incluir la partida recién jugada si se ha superado al menos la décima mejor puntuación.

NOTA: El código fuente y los documentos de este proyecto se encuentran accesibles públicamente en la página web de GitHub.com con el nombre "Gravity-DS": Enlace

5. Conclusiones

La elaboración de este proyecto nos ha reportado grandes beneficios en materia de conocimiento. Una de las primeras conclusiones que sacamos es que la programación de la entrada/salida de un computador no es, ni mucho menos, un enigma irresoluble para nosotros.

Hemos aprendido cómo funcionan los controladores de los periféricos y cómo se pueden adecuar a las necesidades específicas de cada máquina particular en momentos concretos, si bien la versión sobre la que hemos trabajado (Nintendo DS) es más simple que una máquina tan común hoy en día como es el PC. Hemos visto que los computadores poseen diferentes formas de interactuar con el usuario y se pueden gestionar de distintas maneras (encuesta, interrupción...), teniendo que elegir nosotros la más adecuada a nuestro proyecto.

Sobre la arquitectura propia de la Nintendo DS concluimos que es una arquitectura sencilla, pues está basada en gran parte en la arquitectura de su predecesora, la GameBoy Advance, y por lo tanto no es de lo más moderna. Ha sido estudiada en profundidad por muchos desarrolladores, lo que nos ofrece ventaja, ya que existe mucho material explicativo sobre el desarrollo de *homebrew* para NDS en la red. La inexistencia de sistema operativo nos parece un acierto, favoreciendo la libertad para idear y crear con la máquina.

Otra conclusión importante es la sencillez con la que se puede hacer un programa para una plataforma distinta a la que se está usando. No imaginábamos que con sólo un par de clicks pudiésemos compilar un programa para Nintendo DS al igual que compilamos para el PC. Nos adentramos también en un mundo nuevo, el *cross compiling*.

Por último, pero no por ello menos importante, como grupo hemos salido reforzados de esta experiencia. El proyecto nos ha servido para enfrentarnos a problemas reales que podremos encontrarnos en el mundo laboral. El reto de trabajar en equipo no ha podido con nosotros, y además nos atrevemos a decir que lo hemos superado con nota. La comunicación y el esfuerzo de

todos y cada uno de los componentes del equipo han sido la clave para llevar este trabajo adelante.

El resultado del proyecto (el videojuego) ha cumplido las expectativas que nos habíamos marcado. Si bien han quedado unos pocos aspectos que se podrían pulir, como el sistema de detección de colisiones o una canción de fondo mientras se juega, creemos que hemos superado los objetivos mínimos y controlamos el subsistema de entrada/salida, que no se debe olvidar que era el principal objetivo de este proyecto.

6. Bibliografía

[1] Libnds Documentation, https://libnds.devkitpro.org/index.html, 2012. (Última visita mayo del 2012).

[2] Tom Preston-Werner, Chris Wanstrath, and PJHyett, https://github.com, 2012. (Última visita mayo del 2012).

[3] Nintendo DS 2D HD Tutorials, http://www.liranuna.com/nds-2d-tuts/, Lesson 5: Extended Palettes and Rotation , LiraNuna 2012. (Última visita mayo del 2012).

[4] Biblioteca Central – Facultad de Ciencias Físicas y Matemáticas – Universidad de Chile "LISTADO DE EJEMPLOS DE REFERENCIAS BIBLIOGRÁFICAS", http://www.cec.uchile.cl/~ict/referencias.html. (Última visita, mayo de 2012).

[5] Nintendo DS 2D HD Tutorials, http://www.liranuna.com/nds-2d-tuts/, Lesson 4: Multiple Backgrounds and scrolling, LiraNuna 2012. (Última visita mayo del 2012).

[6] A Livingston Technologies Production, http://cooltext.com , 2012. (Última visita mayo del 2012).

[7] Index of /~montoliu/docs/pfc, Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.24 with Suhosin-Patch mod_ssl/2.2.8 OpenSSL/0.9.8g Server at www.vision.uji.es Port 80, http://www.vision.uji.es/~montoliu/docs/pfc/, 2012. (Última visita mayo del 2012).