Universidad
de La Laguna

Escuela Superior de
Ingeniería y Tecnología
Sección de Ingeniería Informática

Trabajo de Fin de Grado

Implementación de video juegos para neurorrehabilitación a través de eye-tracking

Implementation of video games for neurorehabilitation through eye-tracking

Manuel Alejandro Rodríguez Santana

La Laguna, 3 de Junio de 2016

Dña. **Silvia Alayón Miranda**, con N.I.F. 43.812.596-B profesora Contratada Doctor adscrito al Departamento de Ingeniería Informática de Sistemas de la Universidad de La Laguna, como tutor.

D. **Francisco José Fumero Batista**, con N.I.F. 45.731.321-F, personal investigador en formación adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor.

D. **Cristián Modroño Pascual**, con N.I.F. 51.062.048-R investigador del Grupo de Neuroquímica y Neuroimagen de la Universidad de La Laguna

C E R T I F I C A (N)

Que la presente memoria titulada:

"Implementación de video juegos para neurorrehabilitación a través de eye-tracking."

ha sido realizada bajo su dirección por D. **Manuel Alejandro Rodríguez Santana**, con N.I.F. 78.552.972-Z.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 3 de Junio de 2016.

Agradecimientos

Quiero agradecer a mis tutores del TFG la paciencia mostrada y su predisposición siempre a ayudar en todo lo que pudieran. Tanto en lo académicamente necesario, como en lo personal.

Licencia

Resumen

El objetivo del presente TFG es el desarrollo de un video juego para su uso en neurorrehabilitación, que pueda ser controlado manualmente, con un ratón, y visualmente, por medio de un dispositivo de eye-tracking.

Para la recuperación de las funciones motoras de los miembros superiores es esencial incrementar la actividad en las regiones motoras del cerebro. Diversos resultados previos de investigación básica han demostrado que los circuitos cerebrales para los movimientos oculares y los de los miembros superiores están sobrepuestos y distribuidos a lo largo del cerebro. Por este motivo, se desea comprobar si la estimulación de estas zonas cerebrales por medio de movimientos oculares puede mejorar las habilidades motoras del paciente.

Se ha intentado diseñar un juego lo más divertido, intuitivo y simple posible, con una componente adictiva para facilitar las tareas de rehabilitación. Además, en el diseño del juego también se han tenido en cuenta las limitaciones existentes impuestas por el uso del dispositivo de eye-tracking.

Este TFG se ha realizado en colaboración con el Grupo de Neuroquímica y Neuroimagen de la Universidad de La Laguna, que serán los usuarios finales del juego diseñado.

Palabras clave: video juego, eye-tracker, neurorrehabilitación, juego serio.

Abstract

The main objective of this final undergraduate project is to implement a video game for neurorehabilitation, which may be controlled with the hand, with a mouse, and with the eyes using an eye-tracker device.

For recovering the motor functions of the upper extremities is essential to increase the activity of the motor areas of the brain. Previous research works have shown that brain circuits responsible for the eyes' and upper extremities' movements are overlapped and distributed along the brain. For this reason, the objective of this project is to test whether the stimulation of these brain areas through eyes' movements can improve the motor skills.

We have tried to design a funny, intuitive and simple video game, with an addictive component to make easier the rehabilitation tasks. In addition, the limitations due to the use of an eye-tracker device have been taken into consideration.

This project has been carried out in collaboration with the Group of Neurochemistry and Neuroimaging at Universidad de la Laguna. They will be the final users of the game.

Keywords: videogame, eye-tracker, neurorehabilitation, serious game.

Índice General

Índice de figuras

Índice de tablas

1. Introducción

La **neurorrehabilitación** es un proceso médico asistencial complejo, que tiene como objetivo restituir, minimizar y/o compensar en la medida de lo posible las alteraciones funcionales aparecidas en una persona afectada por una discapacidad grave, como consecuencia de una lesión del sistema nervioso central [1].

El daño cerebral es la principal causa de discapacidad del adulto en los países de nuestro entorno. Una de las principales consecuencias negativas derivadas del daño cerebral son los déficits motores, presentes en el 70% de los ictus. En concreto, las complicaciones que afectan a los miembros superiores están entre las más incapacitantes, llegando a imposibilitar en muchas ocasiones las acciones de la vida cotidiana.

El grupo de médicos colaboradores en el proyecto, pertenecientes al Grupo de Neuroquímica y Neuroimagen de la Universidad de La Laguna, ha realizado algunos estudios previos que demuestran que la parte del cerebro que se activa cuando se realiza algún movimiento de la mano es muy similar a la que se activa cuando se mueven los ojos [2]. Este hecho, que podría parecer sorprendente, no lo es tanto si se tiene en cuenta que los movimientos de las manos y los ojos están muy acoplados en la vida diaria.

En la figura 1 se puede observar, a través de imágenes de resonancia magnética funcional, la actividad cerebral de un paciente cuando "controla" un objeto virtual con los ojos y/o la mano.

Se parte de la hipótesis siguiente: a través del movimiento ocular se pueden estimular las zonas del cerebro implicadas en el movimiento de las manos, y este hecho podría ayudar en el proceso de rehabilitación de un paciente con dificultades motoras.

Por lo tanto, el objetivo del presente TFG es el desarrollo de un juego serio, concretamente un video juego, que permita al paciente estimular las zonas cerebrales implicadas en los movimientos de las extremidades superiores.

Figura 1. Activación de zonas cerebrales motoras cuando el paciente "controla" un objeto virtual con los ojos y/o la mano. Estudio realizado por el Grupo de Neuroquímica y Neuroimagen de la ULL con resonancia magnética funcional.

Un **video juego** es un juego electrónico en el que una o más personas interactúan, por medio de un controlador, con un dispositivo dotado de imágenes de vídeo. Normalmente se diseñan con fines lúdicos y suelen ser controlados de modo manual.

En el caso que nos ocupa, además de este control manual, será necesario integrar un control visual, para poder ejercitar los movimientos oculares. Para ello se utilizará un eye-tracker, que es un dispositivo capaz de localizar el punto de la pantalla que está mirando el usuario en cada momento.

2. Objetivos

El objetivo del presente TFG es el desarrollo de un video juego para su uso en neurorrehabilitación, que pueda ser controlado manualmente, con un ratón, y visualmente, por medio de un dispositivo de eye-tracking.

Este video juego debe cumplir los siguientes requisitos:

- Debe ser sencillo e intuitivo: todas las interfaces deben de ser lo más claras, sencillas e intuitivas posibles, para que cualquier paciente pueda usarlo sin ningún tipo de problema y sin necesitar ningún tipo de manual o ayuda.
- Debe ser adictivo: los procesos de rehabilitación son muy lentos, y deben ser constantes, por lo tanto, se debe intentar que el juego atrape la atención del usuario, que lo "enganche", que haga que desee jugar más y mejorar, para que no resulte una carga muy grande el hecho de pasar mucho tiempo jugando.
- Debe ser visualmente agradable: como el usuario debe pasar muchas horas mirando la pantalla, los gráficos deben ser sencillos y agradables, no sobrecargados, que no cansen la vista.
- Se deben guardar las posiciones que el usuario mira en cada momento, registradas con el eye-tracker, para permitir al equipo médico que pueda analizarlas posteriormente. Sin embargo, esta operación no es necesaria siempre, ya que habrá muchas partidas de prueba o de calibración antes de empezar con los experimentos. Por lo tanto, se debe poder seleccionar cuándo se guardan los datos y cuándo no.
- El juego debe permitir al paciente controlar el movimiento de un "objeto" con la vista en un espacio 2D. Además, debe tener la impresión de que lo está moviendo, y no de que sencillamente hay un punto que sigue con la mirada.

- Se puede prescindir del sonido en el juego, ya que el paciente no podrá escuchar nada mientras esté jugando dentro de la resonancia magnética.

3. Elección del juego

En el proceso de selección del juego se han tenido en cuenta diversos factores:

- Los pacientes deberán jugar varias veces al día, y durante ratos largos, en intervalos de 30 minutos a 1 hora. Por lo tanto, el juego debe ser divertido, re-jugable y no muy sobrecargado visualmente, para que no se le canse demasiado la vista al paciente.
- Se debe poder controlar el juego solamente con la vista, con ningún otro dispositivo más al mismo tiempo. Por lo tanto, el juego no puede ser muy complejo, ni con muchos objetos en pantalla. Debe ser algo muy fácil de controlar, sin muchas opciones, o buscar la manera que esas opciones se hagan automáticamente, para que al paciente no le suponga demasiado esfuerzo jugar solamente con la vista.
- El paciente debe tener la sensación de que está moviendo algún "objeto" con la vista en un espacio de dos dimensiones, para que sea un juego válido para neurorrehabilitación.
- El escaso tiempo con el que se cuenta para el desarrollo del juego, unido a la falta de experiencia inicial del estudiante que desarrolla el TFG, también es un factor a tener en cuenta en la elección del mismo.
- El juego seleccionado le debe gustar a cualquier persona. El perfil de usuario es muy variado. Puede que lo usen personas mayores que no están acostumbradas a casi ningún video juego, o, por el contrario, personas jóvenes, que están más familiarizadas con este tipo de actividad.

Considerando todas estas condiciones, se pensaron diversas posibilidades. Tras consultarlas con el cotutor externo Cristian Modroño, se acabó tomando la decisión de que el juego que se debía implementar no debía ser novedoso. Sería uno que ya existiera y que fuera muy popular. Para ello, se estudiaron muchos juegos antiguos famosos, conocidos por una gran cantidad de personas.

Finalmente, se llegó a la conclusión de que una buena opción sería el **Space Invaders**. Es un juego arcade diseñado por Toshihiro Nishikado y lanzado al mercado por primera vez en 1978. Su éxito fue tal en la época, que dio lugar a todo tipo de merchandising. Años después se siguieron sacando versiones del mismo, con prácticamente la misma jugabilidad, pero con mejores gráficos [3].

Por lo tanto, la elección de este juego ofrece la posibilidad de que la mayoría de personas lo conozca. Ya sea porque vivieron su momento de salida o porque lo han jugado alguna vez, en alguna de las múltiples plataformas que ha sacado su propia versión.

Llegados a este punto, lo siguiente a determinar fue el aspecto gráfico que tendría el juego. Se tomó la decisión de que su aspecto sería lo más parecido posible al que tenía el juego cuando fue lanzado por Atari (figura 2).

Figura 2. Space Invaders Atari

Este estilo elegido es muy simple visualmente, pero a la vez es bonito, y conserva la esencia del Space Invaders original. Además, este juego cosechó un éxito enorme en su época, por lo que seguramente muchos de los pacientes lo conocerán y les gustará volver a jugarlo.

3.1 Explicación del juego

La trama del juego es muy sencilla, el usuario maneja un cañón que puede mover de izquierda a derecha en un espacio de dos dimensiones. El objetivo consiste en destruir a todos los extraterrestres que están intentando descender a la Tierra. Hay tres tipos de enemigos distintos.

Durante una partida, el usuario deberá ir eliminando a todos los marcianos que aparecen en pantalla, antes de que estos lleguen abajo, a donde se encuentra el cañón que controla. Si no lo logra y estos tocan a su cañón, habrá perdido instantáneamente. Luego, otra forma de perder, es que los extraterrestres golpeen al jugador en tres ocasiones con sus disparos. Cuando se recibe un disparo, se obtiene un par de segundos de invulnerabilidad y se sigue jugando.

En este juego no hay forma posible de ganar, se deberá jugar hasta perder. Una vez se destruyan a todos los enemigos en pantalla, se volverán a regenerar todos en la parte superior de la pantalla, como al comienzo de la partida. Sin embargo, estos aparecerán moviéndose más rápido y disparando en más ocasiones. Este ciclo se repetirá hasta perder de alguna de las dos formas descritas anteriormente.

Por lo tanto, el objetivo real del juego, consiste en matar la mayoría de marcianos posibles para obtener la mejor puntuación. Consiguiendo así aparecer entre los primeros puestos de la tabla de puntuación, a la cual se puede acceder desde el menú principal del juego.

4. Herramientas

En este apartado se comentarán las distintas herramientas que se han estudiado antes de empezar con la implementación del juego y cuáles se han seleccionado finalmente.

4.1 Herramientas de desarrollo gráfico

Hay dos formas de empezar el desarrollo de un juego:

1. Pensar previamente una historia, que será el hilo conductor del juego, y decidir posteriormente su jugabilidad. Esta primera forma se emplea en aquellos casos donde lo más importante es la trama que tendrá el video juego, por encima de la jugabilidad.
2. Centrarse en la jugabilidad y añadir posteriormente una historia breve, simplemente para que dé pie al juego. A veces, incluso sin historia, como puede ser el caso del famoso Tetris. Este segundo caso cada día se usa menos, últimamente es más común que todos los juegos tengan alguna trama argumental.

En el caso de este TFG se ha seguido la segunda estrategia, debido a que el juego seleccionado ya existe y no tiene ningún tipo de historia asociada. Lo único importante es que tenga una jugabilidad muy entretenida, con el objetivo de que los usuarios jueguen una y otra vez, cansándose lo menos posible.

Existen múltiples opciones a la hora de elegir las herramientas de implementación. A continuación, se explicará brevemente cuáles se estudiaron y por qué se eligió finalmente el motor gráfico Unity.

4.1.1 Unreal Engine

Unreal Engine es un motor gráfico desarrollado por la empresa *Epic Games* en 1998, siendo la base de muchos juegos famosos de la época, como *Unreal Tournament,* por ejemplo [4]. Este motor ofrece un gran abanico de herramientas para desarrollar video juegos, desde juegos simples 2D para móviles, hasta desarrollo de proyectos en realidad virtual.

A la hora de redactar este documento, se encontraba en la versión 4.11, y era una herramienta muy usada, tanto por profesionales como por principiantes. Esto es debido a que tiene una curva de aprendizaje bastante buena, lo que es algo atractivo para nuevos usuarios. Otro factor que también influye mucho en que sea un motor muy popular, es que está programado en C++, un lenguaje mediante el cual se pueden realizar programas muy optimizados. Además, soporta OpenGL, DirectX 11, DirectX 12 y es compatible con varias plataformas (Windows, GNU/Linux, MAC OS) y algunas de las consolas más importantes del momento (Xbox One y Play Station 4) [5].

En cuanto al precio de esta herramienta, desde la versión 4 la política de la empresa ha cambiado. Ahora el software es totalmente gratuito durante la fase de desarrollo, hasta que se obtengan los primeros 3000 US$ de ganancias. A partir de ese momento, se deberá pagar el 5% de las ganancias que se obtengan con los productos desarrollados con ese motor [5].

Sin embargo, a pesar de todo esto, ésta no ha sido la herramienta que se ha empleado en el desarrollo de este TFG. El único motivo que se ha tenido en cuenta para no usarla es el tiempo limitado del que se disponía para la implementación del juego. Por lo tanto, se ha preferido emplear otra herramienta muy parecida, Unity, que se describirá más adelante.

4.1.2 Frameworks

Se pueden emplear herramientas de desarrollo integral, como las mencionados anteriormente, o también se puede emplear algún framework de librerías gráficas para crear un video juego. Sin embargo, esta elección conllevaría mucho más trabajo que la de usar un motor gráfico, y los resultados suelen ser bastante peores.

Se estudiaron varias librerías, concretamente *SDL* [6] o *libGDX* [7], pero no se escogió ninguna por el motivo expuesto anteriormente: el tiempo del que se dispone para realizar este desarrollo es muy limitado, y además el resultado final debe ser agradable a la vista. Obtener algo así con librerías gráficas, con las que no estamos familiarizados, es algo que llevaría demasiado tiempo.

4.1.3 Unity

Unity es un motor gráfico desarrollado por *Unity Technologies*, con fecha de lanzamiento el 30 de mayo de 2005 [8]. Al igual que el Unreal Engine, ofrece muchísimas herramientas, desde para hacer algo muy simple en 2D, hasta para trabajar con realidad virtual.

La versión en la que se encuentra actualmente es la 5.3.4. Es un proyecto que se encuentra en constante actualización y que cuenta con una comunidad muy activa. Además, tiene las siguientes características:

- Multiplataforma: permite crear juegos para Windows, MAC OS, Linux, Xbox 360, Xbox One, Play Station 3, Play Station 4, Play Station Vita, Wii, Wii U, iPad, iPhone, Android y Windows Phone [9].
- Buena curva de aprendizaje, sencillo de aprender, pero sin quitarle potencia.
- API bien documentada [9].
- Foro con usuarios y administradores muy activos para poder consultar dudas [9].

Los lenguajes que soportan este motor son los siguientes [9]:

- JavaScript.
- Boo (muy similar a python)
- C#

En cuanto al precio de Unity, existen dos versiones. La versión "Unity Professional", que cuesta $1500 si se desea tener una licencia permanente, o bien $75 mensuales. Esta versión ofrece algunas herramientas más que la otra, y todo lo que se desarrolle bajo esta licencia será propiedad completa del creador, independientemente del beneficio que obtenga con ello. La otra versión es la "Unity Personal Edition", que es totalmente gratuita. El beneficio obtenido con ella será del desarrollador hasta que se sobrepase los $100 000, lo que obliga a comprar la versión profesional [10].

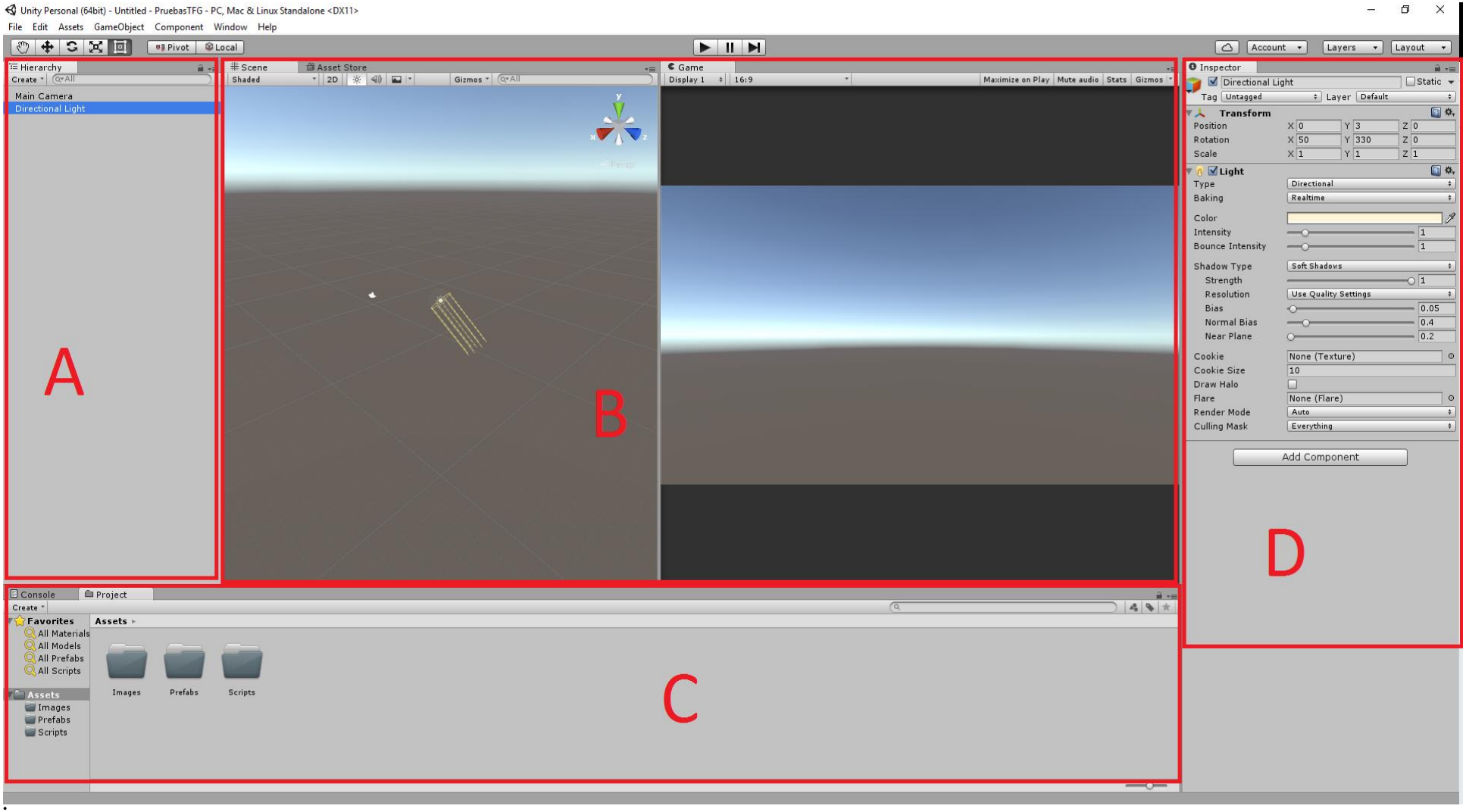

Figura 3. Ventana principal de Unity

La figura 3 muestra la ventana principal de Unity, en la cual se diferencian principalmente cuatro zonas:

Zona A: muestra los objetos que existen dentro de la escena actual. En el caso de la imagen de la figura 3, sólo hay una cámara y una luz direccional.

Zona B: se encuentra dividida en dos partes:

- En la parte izquierda se encuentra la escena sobre la que trabaja el desarrollador.
- En la parte derecha se observa la pantalla de juego, que sólo se activa al darle a play. Esta pantalla mostraría el resultado final de la aplicación implementada.

Zona C: muestra la consola de ejecución si se tiene seleccionada la pestaña "Console". En cambio, si se tiene seleccionada la pestaña "Project", como se ve en la figura 3, se muestra el directorio del proyecto sobre el que se trabaja. Aquí aparecen todos los ficheros, que pueden ser arrastrados a la escena del juego sin problemas, convirtiendo este motor en uno muy simple e intuitivo.

Zona D: en esta zona se pueden observar todos los scripts y las características que tiene el objeto seleccionado en la zona A. En la figura 3 está seleccionada la luz direccional y se puede ver que tiene dos elementos asignados:

- "Transform": elemento que tiene cualquier objeto de Unity. Define la posición en el espacio del objeto, su rotación y su escala.
- "Light": es el script que produce la luz en la escena. Además, se puede configurar para adaptar la luz a las necesidades del desarrollador.

Además de estas cuatro zonas, en la parte superior de la figura 3 se pueden observar las opciones típicas de cualquier programa, y una caja de herramientas muy básicas, para moverse por la escena o para mover un objeto.

Ésta ha sido la herramienta que se ha escogido para el desarrollado de este TFG, debido a que tiene una buena curva de aprendizaje, ya se había usado para la asignatura de *Interfaces Inteligentes*, es gratuita, y se puede programar en C# (que es uno de los lenguajes que soporta el eye-tracker que

se utiliza en este proyecto). Por todo esto consideramos que permitirá agilizar en gran medida la implementación del juego seleccionado.

4.2 IDE (Integrated Drive Electronics)

En cuanto al IDE que se empleará en este trabajo, sólo se ha tenido en cuenta que fuera gratuito, que ya se conociera y que se pudiera integrar con Unity. Por lo tanto, se han estudiado dos.

El primero ha sido el "MonoDevelop" [11], que incluye lo siguiente:

- Manejo de clases.
- Ayuda incorporada.
- Completamiento de código.
- Diseñador de GUI integrado.
- Soporte para proyectos.
- Un depurador integrado desde la versión 2.2 (actualmente se encuentra por la 5.9.5).

El segundo ha sido el "Visual Studio" [12], que incluye lo mismo que el anterior, pero soporta más lenguajes de programación y tiene una interfaz más cuidada. Además de otras muchas funcionalidades más que no tiene MonoDevelop, pero que, para este proyecto, en principio no se usarán.

El que se ha utilizado finalmente es el Visual Studio, por simple estética, por estar acostumbrados a su uso, y por si en el futuro se desea aprender a utilizar las funcionalidades extras que tiene.

5. Dispositivos físicos soportados

En este apartado se explicarán los dispositivos físicos que deben incluirse en el desarrollo del juego para que se pueda utilizar en neurorrehabilitación: el dispositivo de eye-tracker (para permitir jugar con la vista) y un dispositivo común, como por ejemplo un ratón (para permitir jugar manualmente).

5.1 Eye-tracker

El estudio del movimiento de los ojos no es algo moderno, se lleva investigando desde hace más de un siglo. En 1980 se empezaron a utilizar sistemas de seguimiento de ojos para resolver preguntas relacionadas con la interacción persona-computador. Sin embargo, no fue hasta 2006 aproximadamente, cuando se empezaron a producir eye-trackers a un bajo costo, aunque no eran muy precisos [13].

Los primeros que se utilizaron en el mundo empresarial tenían unos precios de $20 000 a $30 000, que es un precio bastante asequible para una gran empresa, pero no para un usuario particular [13]. No obstante, desde hace algunos años, ya hay empresas que han sacado dispositivos de seguimiento ocular a un precio mucho más razonable para cualquier persona. Un ejemplo de esto, es el que se usará en este proyecto, el de TheEyeTribe [14], que cuesta $99 (aunque, actualmente, están vendiendo otra versión del producto mejorada por $199).

Esta reducción de los precios es otro de los motivos por el cual la investigación que hay detrás de este trabajo se lleva a cabo. Gracias a estos precios más asequibles, cualquier paciente de neurorrehabilitación podría tener su propio dispositivo y realizar su rehabilitación en su casa todos los días sin tener que ir al hospital.

TheEyeTribe, es una empresa danesa “startup“, fundada por cuatro personas en 2011 [14, 15], que produce sistemas de eye-tracking y el software para usarlos. Las características del modelo de $99, que se usa en este TFG, son:

- Ratio de muestreo: de 30 Hz a 60 Hz.
- Latencia: menos de 20 ms a 60 Hz.
- Precisión: 0. 5º de media.
- Calibración: formato de 5, 9 y 12 puntos.
- Máximo tamaño de pantalla soportado: 24 pulgadas.
- Área de funcionamiento: 40cm x 30cm a 65 cm de distancia.
- Lenguajes que soporta la API/SDK: C#, java y C++.
- Peso: 70 g.
- Dimensiones: 20cm x 1.9cm x 1.9cm.
- Necesita ser enchufado a un puerto USB 3.0, en cualquier otro puerto no funcionará, debido a que ha sido optimizado para este puerto.

Entonces, cualquier persona con un ordenador con un puerto USB 3.0 y este eye-tracker, podrá utilizar el juego implementado en su casa sin problema.

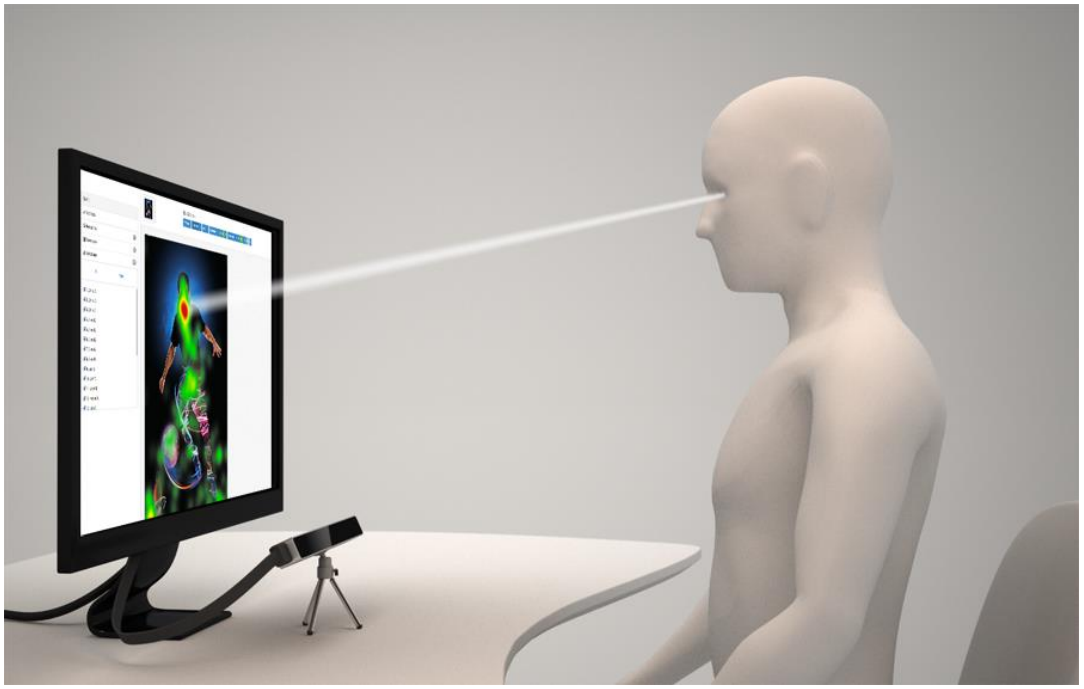

Figura 4. Ejemplo de uso del eye-tracker empleado en este proyecto.

En la figura 4, se puede ver lo sencillo que será para el usuario utilizar este eye-tracker en su casa. Simplemente tiene que enchufarlo y colocarlo en frente de la pantalla. En la figura 5 se puede observar mejor lo pequeño que es.

Figura 5. TheEyeTribe.

5.2 Dispositivo manual

El otro dispositivo que debe soportar el juego desarrollado en este proyecto es el más común hoy en día, un ratón. Para que se puedan hacer pruebas en la investigación que se está llevando a cabo, el paciente debe poder jugar también con la mano. Posteriormente, el equipo médico hará comparaciones entre los estados de activación del cerebro del paciente jugando con el eye-tracker y con el ratón.

6. Juego implementado

6.1 Implementación

Como el juego que se debe desarrollar en este TFG ya existe, nuestros esfuerzos se han enfocado en adaptar su jugabilidad a los requerimientos de las tareas de neurorrehabilitación.

El principal objetivo es que se pueda jugar sólo con la mirada. Entonces, las dos funciones principales que se han tenido en cuenta son:

- Mover al personaje, el cañón que se menciona en la sección 3.1 (Explicación del juego), con la vista. El cañón debe desplazarse a la posición donde esté mirando el usuario
- Controlar el disparo del cañón. Esta funcionalidad es más compleja de implementar con un eye-tracker, ya que el usuario no puede estar mirando a dos sitios a la vez. Cuando el dispositivo no detecta un ojo devuelve un error.

Finalmente, la decisión que se tomó antes de empezar con el desarrollo fue que el cañón disparara automáticamente. El cañón que controla el paciente, estará siempre disparando, con un intervalo de tiempo constante entre disparo y disparo. De esta forma, el paciente sólo se tendrá que preocupar de mover su nave con la mirada, que es el objetivo final que tiene este juego.

El juego fue desarrollado con el motor gráfico Unity, que se menciona en la sección 4.1.3 (Unity). El lenguaje empleado fue C#, debido a que es uno de los que soporta el eye-tracker usado en ese TFG, mencionado en la sección 5.1 (Ete-tracker). El método más importante de todo el código se encuentra en el apéndice A. El resto del código del proyecto se encuentra en el repositorio GitHub que se indica en el apéndice B.

6.2 Resultado

A continuación, se mostrarán imágenes de cómo quedó el juego tras la implementación.

6.2.1 Pantalla de elección de dispositivo de control y de calibración

En la figura 6 se muestra la primera escena del juego. En esta escena se puede hacer lo siguiente:

- Salir, con el botón que hay arriba a la izquierda.
- Seleccionar si se quiere que se guarden o no las posiciones a las que mirará el usuario. Con "Save Data", que aparece arriba a la derecha.
- Seleccionar con qué dispositivo se quiere jugar, en el panel central:
    - "Start Game (mouse)", para que se controle con el ratón, un dispositivo manual.
    - "Reconnect to server", para que se controle con la vista. Sin embargo, para que se pueda hacer esto, se debe tener conectado un eye-tracker y el servidor necesario arrancado.

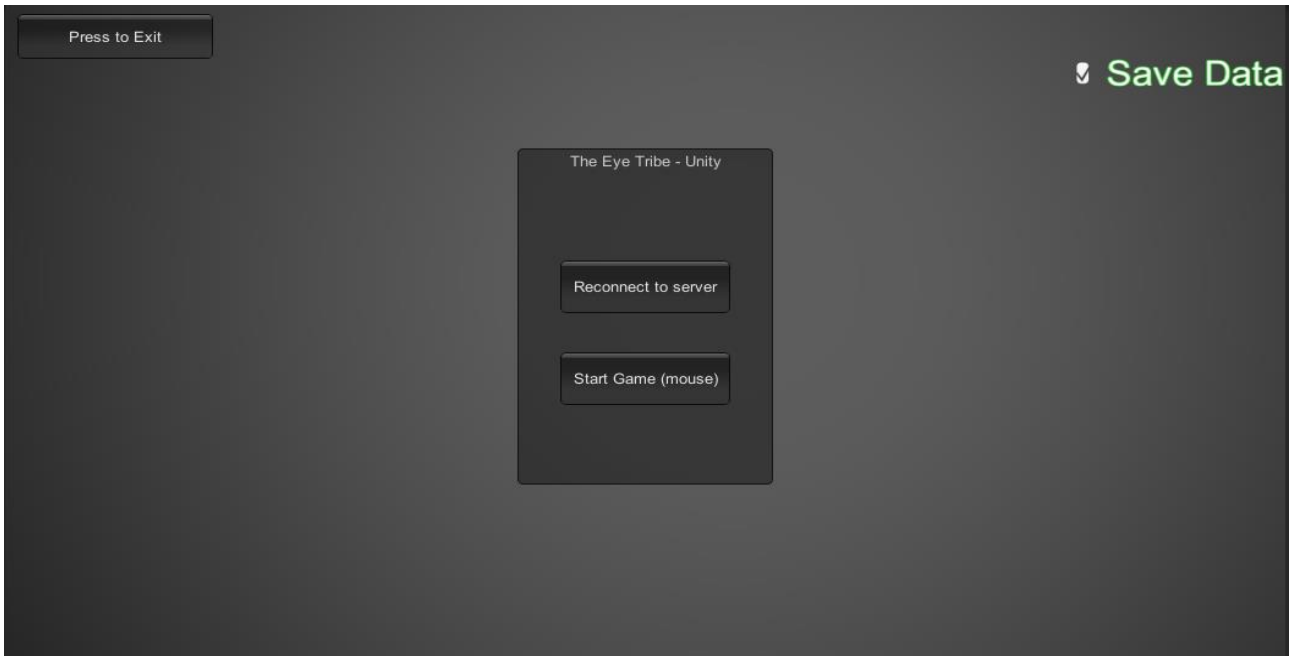

Figura 6. Selección de dispositivo

Esta escena está pensada solamente para los médicos, es la que se verá antes de ponerle el juego al paciente y que éste comience a jugar. Por lo tanto, esta escena sólo puede ser controlada con el ratón.

En la figura 7 se observa la escena de calibración, que se ejecuta siempre que se va a utilizar el control con la vista. En esta imagen hay un punto en la mitad de la pantalla al cual se debe mirar para la calibración del eye-tracker. Este punto permanece unos pocos segundos en cada posición de la pantalla y va cambiando.

Figura 7. Calibración

Para la calibración, se le pasa las coordenadas del punto en la pantalla al SDK del TheEyeTribe y éste realiza los cálculos según la posición de los ojos del usuario. La calibración que hay en este juego es de 9 puntos, pero existen versiones con más puntos, como por ejemplo una de 12 puntos.

Después de este paso, si el usuario ha seguido bien los puntos durante todo el proceso, la calibración terminará con un buen resultado, y las posiciones que devolverá de la vista serán bastante precisas, siempre y cuando el usuario no mueva la cabeza.

6.2.2 Menú Principal

En la figura 8 se muestra el menú principal del juego, que ya será controlado con el dispositivo que haya sido seleccionado en la escena anterior. Aquí se puede ver lo siguiente:

- Los puntos que se consiguen al eliminar un enemigo. Además, también se observa que hay un enemigo "secreto", del cual no se sabe nada.
- Botón "PLAY", para iniciar el juego.
- Botón "POINTS", que muestra otra escena con los resultados obtenidos en partidas anteriores por cualquier persona.
- Botón "EXIT", para cerrar el juego.

Figura 8. Menú Principal

6.2.3 Puntuaciones

En la figura 9 se muestra la escena donde se ven las puntuaciones y los siguientes botones:

- “BACK”, mediante el cual se vuelve a la escena anterior, que es el menú principal.
- “DELETE”, que sirve para borrar todas las puntuaciones que hubieran guardadas.

Además, en la imagen, también se muestra que se puede omitir el nombre del paciente, por si éste no quiere identificarse.

Figura 9. Puntuaciones

En el caso de que hubiera más puntuaciones de las que caben en pantalla, se generará automáticamente un scroll hacia abajo. Consiguiendo así que todo el mundo pueda ver su puntuación. Otro punto importante a destacar es que un mismo paciente puede aparecer varias veces. Así puede ver su mejoría en el tiempo, según vaya jugando más veces, lo que puede ser un aspecto motivador para seguir jugando: el intentar superarse a sí mismo.

6.2.4 Juego

La figura 10 muestra la escena más importante, la del juego en sí mismo, que se comentará por partes:

- En la parte superior aparece la información siguiente:
    - La puntuación que lleva hasta el momento en su partida el jugador.
    - Las vidas que le quedan al jugador, que como ya se comentó en la explicación del juego, hay un máximo de tres.
- En la parte central se ejecuta el juego. Aquí se encuentran los enemigos, que aparecen separados por filas organizadas con diferentes tipos de enemigos. Los enemigos que se encuentran en la parte más alta otorgan más puntos por eliminarlos que los que se encuentran en la parte más baja. Esto es debido a que son más difíciles de alcanzar. Todos se mueven hacia la misma dirección. Cuando uno de ellos choca contra el borde de la pantalla, todos descienden un poco y se comienzan a desplazar hacia el otro lado. Este ciclo se repite hasta que los enemigos alcancen al jugador o hasta que el jugador los elimine a todos.
- En la parte inferior están las barreras con las que cuenta el jugador para defenderse de los disparos enemigos. Éstas se van destruyendo poco a poco con cada disparo, ya sea del enemigo o propio. Y, por último, se encuentra el cañón que controla el jugador, debajo de las barreras, que sólo se puede mover entre los límites de la pantalla.

6.2.5 Introducir nombre

Esta es la última escena que conforma el juego, y es donde el usuario puede introducir su nombre. Esta escena aparece solamente cuando el usuario ha perdido. Hay 2 formas de introducir el nombre. La primera es con el teclado, como se muestra en la figura 11.

En esta primera escena, el usuario sencillamente pulsaría en el teclado su nombre, e iría apareciendo lo que escribe en el recuadro “Enter your name...”. La segunda forma que hay para introducir el nombre es con la vista. La figura

12 aparece cuando se pulsa (o se mira) sobre el icono de teclado que hay en la figura 11.

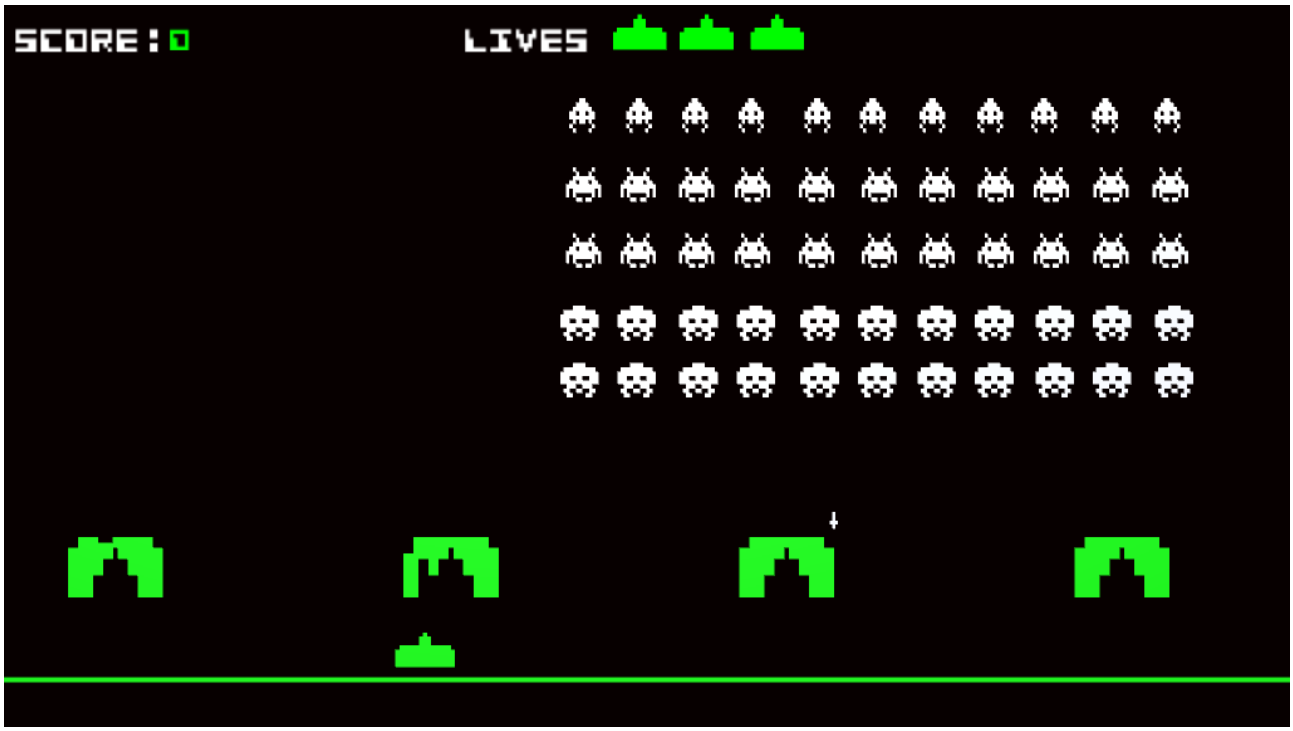

Figura 10. Juego

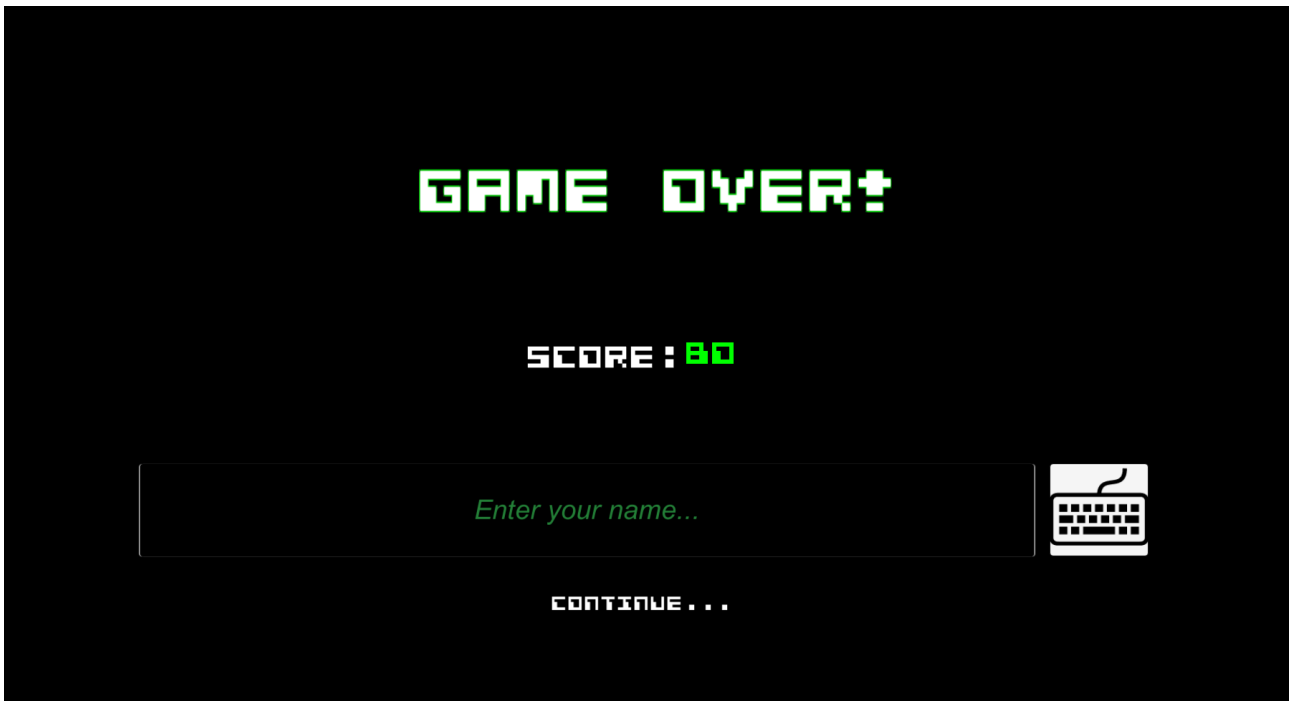

Figura 11. Introducir nombre con teclado manual.

Figura 12. Introducir nombre con teclado sensible a la vista

En este teclado, el usuario solo tendrá que mirar a las teclas, durante un tiempo pre-establecido (para que no seleccione ninguna tecla sin querer), y se escribirá arriba la letra mirada. Luego, al mirar a "ENTER", se volverá a la escena de la figura 11, con el nombre que se haya escrito en ésta.

Finalmente, una vez se ha terminado de introducir el nombre de la forma que sea más sencilla, se pulsa el botón "continue ..." de la figura 11 y se vuelve al menú principal con la puntuación ya guardada.

7. Validación

Después de que se realizaran pruebas con el juego, durante algún tiempo, con 6 personas, se han sacado las siguientes conclusiones:

- Acostumbrarse al juego es muy fácil, con el ratón todo el mundo se adapta al juego sin ningún problema desde el principio. Cuando se emplea el eye-tracker cuesta un poco más, se tarda una media de 5 minutos en acostumbrarse a controlar el juego con la vista.
- El eye-tracker empieza a dejar de ser preciso en pantallas grandes, este efecto se nota a partir de pantallas mayores de 21". También deja de ser preciso cuando la resolución de la pantalla es demasiada elevada, a partir de 1680x1050.
- En la calibración a veces cuesta seguir el punto de calibración del dispositivo, debido a que el punto no se mueve de un sitio a otro, sino que se "teletransporta". Esto se ha implementado de esta manera porque la calibración se lleva a cabo en un hilo aparte del hilo principal de ejecución. Y, en Unity esto es un problema, ya que un objeto sólo puede ser movido en una escena por el hilo principal.
- Pasados unos 15-30 minutos de juego, empieza a aparecer cansancio visual, debido a que jugar con la vista es un proceso al que nadie está acostumbrado. No se han podido hacer pruebas con las mismas personas a lo largo de varios días, por lo tanto, no se sabe todavía si pasadas unas cuantas sesiones de juego, comenzaría a ser menos cansado el jugar sólo con la vista.
- Escribir el nombre del usuario con el teclado visual resulta tedioso muchas veces, ya que el más mínimo movimiento de la cabeza produce errores en el cálculo. Una vez que el jugador ha pasado un rato largo en una partida, en la mayoría de ocasiones, éste mueve la cabeza de forma involuntaria. Esto provoca que las estimaciones de donde se encuentra mirando el paciente no sean muy precisas, aunque para jugar no suponga un grave problema. Para escribir sí lo es, debido a que los botones de las teclas son pequeños, para que quepan todos en pantalla.

8. Conclusiones

8.1 Conclusiones

En este TFG se ha desarrollado un videojuego serio para neurorrehabilitación, controlable con la vista (usando un dispositivo de eye-tracking) o manualmente (con un ratón). El juego se ha implementado lo más atractivo y adictivo posible, para que los usuarios no se aburran rápidamente de él y pueda ser usado con mucha frecuencia, que es el fin perseguido por el equipo médico colaborador en este TFG. En concreto, se ha implementado el juego Space Invaders de Atari usando el motor gráfico Unity.

El estudiante que ha desarrollado este TFG no tenía ningún conocimiento sobre la problemática abordada en este proyecto antes de comenzarlo. Solamente conocía la existencia de los dispositivos eye-tracking, pero nunca los había utilizado. Sobre las herramientas utilizadas en la implementación final, tenía nociones muy básicas de Unity. Por lo tanto, este TFG ha representado una oportunidad para aprender a programar nuevas tecnologías, profundizar en herramientas de las que se sabía muy poco, y colaborar en una investigación médica de envergadura, proponiendo un prototipo de utilidad para los pacientes de neurorrehabilitación del Hospital Universitario de Canarias.

La metodología seguida en este TFG ha sido la siguiente:

- Realizar un estudio inicial de la problemática abordada y de las herramientas disponibles.
- Seleccionar el juego a implementar.
- Seleccionar las herramientas más adecuadas para su implementación.
- Profundizar en la herramienta Unity. Las dudas que surgieron relativas al uso de este motor gráfico fueron sencillas de solucionar en su gran mayoría, gracias a su API bien documentada y a su foro tan activo.
- Diseñar una primera versión del juego para ser jugada manualmente con un ratón.

- Estudiar el dispositivo eye-tracker seleccionado.
- Diseñar la versión final del juego, que contempla el control visual del mismo por medio del eye-tracker. Los problemas surgidos en esta parte fueron un poco más complicados de solucionar, ya que estaban relacionados con errores de falta de precisión del dispositivo.
- Realizar el estudio de validación del juego.
- Entregar el juego desarrollado al equipo médico colaborador, para que empezara a usarlo con pacientes.

8.2 Líneas Futuras

Hemos detectado algunos aspectos que pueden ser mejorados en el juego, y que podrían ser abordados en líneas futuras de desarrollo:

- El rendimiento del juego podría mejorarse si las defensas se implementaran de otro modo, dado que el modo en el que están actualmente no es muy óptimo.
- Sería posible introducir algunos cambios para mejorar la usabilidad del juego. Por ejemplo, cuando el usuario elimina a todos los enemigos y éstos vuelven a aparecer, el aumento de velocidad podría regularse mejor. En la versión actual el cambio de velocidad quizás sea demasiado brusco.
- Sin embargo, la parte más importante que se debería mejorar de este proyecto, es el dispositivo de eye-tracking utilizado. Con el seleccionado en este TFG el más mínimo movimiento de la cabeza ocasiona errores en sus cálculos, llegando a suponer un verdadero problema si el usuario se mueve de forma involuntaria un par de veces. Se propone utilizar el nuevo modelo de Eyetribe, que, según la empresa, es mucho más robusto y funciona correctamente aunque el usuario mueva la cabeza.

Finalmente, mencionar que si el equipo médico determina que los pacientes mejoran con esta estrategia de neurorrehabilitación, otra línea de desarrollo futura podría ser la implementación de nuevos juegos. De este modo, los pacientes tendrían variedad donde elegir, y las sesiones de rehabilitación, que suelen ser largas y pesadas, podrían ser mucho más amenas.

9. Conclusions

9.1 Conclusions

In this final undergraduate project a serious video game for neurorehabilitation has been developed, which can be controlled by the eyesight (using an eye-tracker device) or with the hands (using a mouse). The game has been implementing as attractive and addictive as possible, to ensure that users do not get bored quickly and play with it during long periods, which is the objective pursued by the medical staff of this project. In particular, the implemented game has been the Space Invaders of Atari, using Unity as development tool.

The student had no prior knowledge about the tackled problem at the beginning of the project. He only knew what an eye-tracker device was, but he had never used one before. Regarding the selected tools in this project, he had some basic knowledge about Unity. For these reasons, this final project has been an opportunity to learn to program new technologies, to delve into unknown tools, and to collaborate in an important medical research, where he has contributed with a useful prototype for the neurorehabilitation patients of the Canary University Hospital.

The methodology of this project has been the following one:

- Development of an initial study of the tackled problem and the available tools.
- Selection of the game.
- Selection of appropriate tools for its implementation.
- Deep study of Unity. Doubts about this tool were easily solved thanks to its well documented API and its very active community forum.
- Design of the first version of the game, which could be controlled by a mouse.
- Study of the selected eye-tracker device.

- Design of the final version of the game, which can be controlled by the eyesight. The problems that appeared in this part were more difficult to solve, because they were related to accuracy errors of the eye-tracker device.
- Validation study of the game.
- Delivery of the developed game to the medical staff, to allow its use with patients.

9.2 Future work

We have detected some aspects that could be improved in the game in the future:

- The performance of the game could be improved implementing the defenses in a different way. Their actual implementation is not optimal.
- We could introduce some changes to improve the game usability. For example, when the user destroys all the enemies, and more enemies appear again, the game speed increases a lot, and this could be better regulated. In the current version of the game, this speed change is maybe too rude.
- However, the most important part that should be improved is the selection of the eye-tracker device. The current one is too sensible to little head movements, causing errors in its calculations. This fact becomes a big problem if the patient moves his head few times. A possible solution could be the use of the new model of Eyetribe, which is more robust in the presence of head movements.

Finally, if the medical staff determines that patients improve their motion problems with this strategy of neurorehabilitation, another future line could be the development of new games. With more available games, rehabilitation sessions, that usually are long and tedious, could be nicer and more entertaining.

10. Presupuesto

El presupuesto de este proyecto se muestra en la tabla 1. Se divide en dos partes: coste del material utilizado y coste de las horas de trabajo invertidas por el ingeniero en el desarrollo del producto.

Descripción	Coste
Eye-tracker	99,00€
Horas trabajadas (300 horas * 20 €/hora)	6.000,00€
Total:	**6.099,00€**

Tabla 1. Presupuesto proyecto

Apéndice A. Controlador

A.1 Método "Vector3 GetPosition ()"

```
/*****************************************************************************
*
* Controller.cs
*
*****************************************************************************
*
* Manuel Alejandro Rodríguez Santana
*
*
* 10/04/2016
*
*
* Método que calcula la posición a donde se encuentra mirando el usuario o donde
* tiene el ratón. Este método es usado en todas las escenas del juego y es llamado
* en cada frame. Además, si se quisiera cambiar de eye-tracker, solamente habría
* que cambiar este método y todo el proyecto seguiría funcionando.
*
*****************************************************************************/

public Vector3 GetPosition ()
{
    Vector3 positionMouse;
    Camera positionCamera = GameObject.Find ("Main Camera").GetComponent<Camera> ();
    if ( positionCamera == null )
    {
        throw new System.ArgumentException ("Camera not found");
     }
    if ( !mouse )
    {
          Point2D gazeCoords =
GazeDataValidator.Instance.GetLastValidSmoothedGazeCoordinates ();
          Point2D gp = UnityGazeUtils.GetGazeCoordsToUnityWindowCoords (gazeCoords);
          positionMouse = new Vector3 ((float)gp.X, (float)gp.Y, 0);
          positionMouse = Camera.main.ScreenToWorldPoint (positionMouse);
```

```
            if ( Points.stateGame.saveData )
                  Points.stateGame.Save ((float)gp.X, (float)gp.Y);
       }

      else
      {
            positionMouse = Camera.main.ScreenToWorldPoint ( new Vector3 (
Input.mousePosition.x, Input.mousePosition. y, positionCamera.nearClipPlane + 1f ));
       }

      return positionMouse;
  }
```

Apéndice B. Github

B.1 Repositorio github con todo el código

Todo el código de este proyecto se encuentra en el siguiente repositorio de github "https://github.com/Manwelanza/SI_eyetribe". Como el repositorio es público, cualquiera con el enlace puede ver todo el contenido.

Todo el código, excepto la parte del eye-tracker, se encuentro dentro de este repositorio en el directorio: "Assets/Scripts". Luego, la parte del código del eye-tracker, se encuentra en el directorio: "Assets/EyeTribe/Plugins/Scripts".

Bibliografía

[1] Neurorrehabilitación. http://www.neuroped.es/que-es-neurorehabilitacion-y-estimulacion-del-neurodesarrollo/.

[2] C. Modroño, J. Plata-Bello, F. Zelaya, S. García, I. Galván, F. Marcano, G. Navarrete, O. Casanova, M.Mas and J.L. González-Mora. Enhancing Sensorimotor Activity by Controlling Virtual Objects with Gaze. *Plos One*, 10(3): e0121562. doi:10.1371/journal.pone.0121562, 2015.

[3] Space Invaders. https://es.wikipedia.org/wiki/Space_Invaders

[4] Unreal Engine. https://es.wikipedia.org/wiki/Unreal_Engine

[5] Epic Games, Unreal Engine 4. https://www.unrealengine.com/what-is-unreal-engine-4

[6] Simple Directmedia Layer (SDL). https://www.libsdl.org/

[7] libGDX. https://libgdx.badlogicgames.com/

[8] Unity (software). https://es.wikipedia.org/wiki/Unity_(software)

[9] Unity3d. https://unity3d.com/es/unity

[10] Unity3d. http://unity3d.com/es/unity/faq

[11] Monodevelop. http://www.monodevelop.com/

[12] Visual Studio. https://www.visualstudio.com/features/unitytools-vs

[13] Y. Hassan Montero, V. Herrero Solana. Eye-Tracking en Interacción Persona-Ordenador, 2007. http://www.nosolousabilidad.com/articulos/eye-tracking.htm

[14] The Eye Tribe. https://theeyetribe.com/

[15] The Eye Tribe. https://en.wikipedia.org/wiki/The_Eye_Tribe