MEMORIA DE PRÁCTICA EXTERNA
ETS DE INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN
UPM

Pablo Parra Iglesias

ÍNDICE DE CONTENIDOS

1. INTRODUCCIÓN:

El objetivo de estas prácticas ha sido la realización de un videojuego mediante el que, utilizando la cámara Kinect, personas con movilidad reducida puedan ejercitarse y rehabilitarse de una forma divertida. De esta forma será necesario emplear sistemas de control basados en gestos, en lugar de la interfaz tradicional basada en teclado y ratón.

Durante los meses que han durado estas prácticas se ha creado distintos ejercicios, que formarán parte de un proyecto de mayor dimensión, puesto que es un proyecto a largo plazo.

Para ello se ha investigado en técnicas de reconocimiento de lenguaje corporal, se ha realizado búsqueda de información científica y se ha estudiado qué tipos de ejercicios podrían convenir para la rehabilitación de personas con movilidad reducida.

Para desarrollar este videojuego, se ha hecho uso del programa Blender, que permite el modelado, iluminación y animación de gráficos tridimensionales. Además, gracias a su motor de juegos (Blender Game Engine) dispone de las capacidades para el desarrollo del videojuego.

En este documento se explicará cómo se ha desarrollado el proyecto, qué tecnologías y medios técnicos se han necesitado para realizarlo, los plazos en los que se han ido completando los objetivos, las competencias adquiridas y posibles mejoras para las personas que sigan con este proyecto en un futuro.

2. INFORMACIÓN DE LA ENTIDAD COLABORADORA

El Centro de Investigación en Tecnologías Software y Sistemas Multimedia para la Sostenibilidad (CITSEM) pertenece al ámbito del I+D+i del Campus Sur de la Universidad Politécnica de Madrid. Tiene como actividad principal aprovechar las tecnologías básicas en las que sus miembros tienen experiencia, tecnologías software y multimedia en un contexto de sostenibilidad.

El CITSEM se ubica en el edificio La Arboleda perteneciente al Campus Sur de la UPM. Cuenta con más de cuarenta investigadores a tiempo completo. Gran parte del personal de este centro forma parte del profesorado o alumnado de la universidad. Está formado por tres grupos de investigación reconocidos por la UPM: Grupo de Tecnología Software y Sistemas (SYST), Grupo de Diseño Electrónico y Microelectrónico (GDEM), y Grupo de Redes y Servicios de Próxima Generación (GRyS).

La investigación está dividida en tres áreas: Redes y Servicios Ubicuos (RSU), Tecnologías de Vídeo e Imagen (TVI) y Tecnologías Software y Servicios (TSS).

La línea de TVI tiene una vertiente tecnológica y otra más relacionada con las aplicaciones. En la vertiente tecnológica se investiga en temas relacionados con la codificación, transmisión, recepción y descodificación de la información de audio y vídeo en los sistemas multimedia.

3. ENMARCAR LAS PRÁCTICAS EN EL CONTEXTO DE LA ENTIDAD

Estas prácticas han consistido en el desarrollo de un videojuego que sirva para que personas con movilidad reducida puedan rehabilitarse de una forma divertida. Para ello se captan sus movimientos con la cámara Kinect y, en función de ellos, se realizan los ejercicios.

Esta labor forma parte de la investigación de tecnologías de realidad aumentada del CITSEM, que sigue esta línea desde hace varios años. Se continuado el trabajo de otros alumnos en semestres anteriores sobre la conexión de Kinect con Blender, gracias al middleware.

En este proyecto ha sido realizado por tres alumnos de la universidad.

El trabajo ha estado supervisado por Martina Eckert, profesora de la UPM, miembro del grupo de investigación GRyS. Cada dos semanas, como mínimo, tenía lugar reunión con el grupo de trabajo y Martina. Se explicaban los avances del proyecto y cómo se habían realizado. Se probaba la funcionalidad y se proponían mejoras.

Mensualmente, se organizaba una reunión con otros alumnos que, en su trabajo de fin de grado, trabajaban en investigación de tecnologías de realidad aumentada: con las gafas Oculus Rift y con la Kinect para la Xbox One (versión mejorada de la Kinect empleada en estas prácticas). En estas reuniones se planteaban los logros obtenidos y problemas encontrados, de forma que todo el grupo estaba al corriente del trabajo de los otros y podía hacer sugerencias u obtener ayuda.

4. OBJETIVOS DE LAS PRÁCTICAS, TAREAS Y ACTIVIDADES REALIZADAS

El objetivo global de estas prácticas era realizar un videojuego que permitiese la rehabilitación de personas con movilidad reducida, cuya jugabilidad dependiese de los movimientos captados por la cámara Kinect.

La idea es crear un videojuego sobre un mundo imaginario formado por diversos climas y distintas localizaciones. Cada una de estas localizaciones formará parte de una escena distinta pero se podrá acceder de una escena a otra fácilmente. En cada localización se podrán realizar algunas actividades para seguir la historia del juego y, sin darse cuenta, el usuario estará realizando una sesión de rehabilitación.

En cada ejercicio habrá un método de medición que evalúa cómo ha finalizado el usuario esa prueba, ya sea con el tiempo empleado en su finalización, la velocidad máxima que se ha registrado en el ejercicio, velocidad media del personaje, número de veces que ha conseguido el objetivo, etc. Todos estos parámetros serán guardados en un fichero de forma que el terapeuta o el usuario puedan ver las evoluciones del paciente.

Al inicio del juego, se pedirá al usuario realizar determinadas acciones para evaluar cuál es su limitación de movimientos. De este modo, automáticamente, el programa adapta los movimientos del personaje durante el juego da tal forma que se realiza una amplificación de movimientos gracias al addon programado en Python.

Al empezar el juego, el personaje tendrá una forma animal, similar a la de un renacuajo. Pero según va avanzando el juego, se produce un proceso de evolución en el personaje. Estas evoluciones producen que el personaje adquiera unas determinados rasgos, como aumento de la longitud de las extremidades, obtención de pulgares oponibles, adquisición de alas para volar, etc. Todo ello le permite realizar nuevas acciones y, por lo tanto, nuevos ejercicios.

Este proyecto es a largo plazo por lo que para estas prácticas se ha centrado en la realización de algunos minijuegos independientes que permitiesen ejercitar distintas partes del cuerpo para la rehabilitación y/o diversión del usuario.

Se realizó un estudio del arte sobre el uso de la Kinect para la realización de ejercicios físicos, principalmente de rehabilitación, como documentación para saber qué tipos de ejercicios convendría incluir en el proyecto. Para ello se realizó una investigación en vídeos de YouTube y publicaciones científicas.

Todas las actividades se han dividido en tres tipos de ejercicios:

- Rehabilitación de extremidades superiores
- Rehabilitación de extremidades inferiores
- Rehabilitación de tronco

Se han realizado tres ejercicios para la rehabilitación de las extremidades superiores. En ellos, el personaje copia los movimientos de brazos y hombros del usuario, captados por la Kinect.

El primer ejercicio consta de una escalera de mano que tiene que ser subida por el personaje. Mientras está agarrado con una mano a la escalera, el otro brazo copia los movimientos del usuario captados a través de la Kinect. Al alcanzar determinada altura, salta una animación en la que el otro brazo alcanza el escalón superior y, posteriormente, este brazo queda enganchado a la escalera y el otro empieza a copiar los movimientos del otro brazo. Y así sucesivamente hasta alcanzar la meta. En todo momento hay un indicador que muestra la distancia que le queda al usuario para alcanzar la meta, al final de la escalera. Al llegar a la cima, salta un mensaje en la que se muestra cuánto tiempo ha tardado el usuario hasta alcanzar la cima.

En el segundo ejercicio el usuario tiene que golpear a unos topos que salen del suelo aleatoriamente. Para ello cuenta con una maza en el brazo derecho. El personaje copia los movimientos de ambos brazos del usuario captados por la Kinect. Solo se considera que el topo ha sido golpeado si se mueve el brazo de la maza a una determinada velocidad. Durante el juego se muestran cuantos topos han sido golpeados. Al finalizar el juego, que será cuando hayan salido ya veinte topos, se muestra un cartel en el que aparecen cuántos topos han sido golpeados, en comparación con los veinte topos que han aparecido.

En el tercer ejercicio el personaje está situado en una barca que tiene que alcanzar una meta. El personaje copia el movimiento de los brazos del usuario. Además, los remos de la barca siguen a los brazos del personaje. Al realizar un movimiento de remo (los dos brazos hacia adelante seguido de los dos brazos hacia atrás) la barca se desplaza una determinada cantidad. En todo momento aparece un letrero que anuncia cuánta distancia queda por recorrer hasta que se alcance la meta. Al finalizar el ejercicio se muestra cuánto tiempo se ha tardado en recorrer y cuál ha sido la velocidad media empleada para llegar hasta el final.

Los otros ejercicios correspondientes a la rehabilitación de extremidades inferiores, superiores y el tronco han sido un ejercicio de baile en el que el usuario tenía que copiar los movimientos que le indicaban al personaje del juego y otro ejercicio en el que el usuario podía volar controlando la subida o bajada mediante el movimiento del tronco y controlando el desplazamiento lateral según la posición de los brazos.

Hubiera sido ideal haber podido enlazar los distintos minijuegos creados mediante una pequeña historia que los enlazara pero ha sido imposible debido a que, al estar el fin de prácticas cerca, se ha preferido aplicar el amplificador de movimientos y poder realizar pruebas con ello.

5. TECNOLOGÍAS Y MEDIOS TÉCNICOS UTILIZADOS

- Blender

Es un software de libre distribución para el modelado, animación y renderizado de gráficos 3D. Tiene su propio motor de juegos en el que permite crear aplicaciones interactivas en tres dimensiones. Su motor de juegos (Blender Game Engine) es una potente herramienta de alto nivel de programación. Blender está construido con Python como base, permitiendo que se un programa ampliable y personalizable. La lógica del juego puede estructurarse mediante bloques lógicos (formados por sensores, controladores y actuadores) aunque también puede controlarse mediante scripts programados en Python, de forma que se puede aumentar las funcionalidades del programa y acceder a objetos y propiedades que no podrían controlables mediante bloques lógicos.

- Kinect

Es un controlador de juego libre y entretenimiento desarrollado para la videoconsola Xbox 360 aunque es compatible para PC. Kinect permite a los usuarios controlar e interactuar con el PC sin necesidad de tener contacto físico con un controlador de videojuegos tradicional, mediante una interfaz natural de usuario que reconoce gestos y movimientos. Consiste en una cámara de vídeo de 640x480 píxeles de resolución, y un sensor de profundidad. Microsoft proporciona un entorno que permite extraer la posición de una o más personas a través de la imagen captada por la cámara, además de un esqueleto con veinte articulaciones.

- KinectOSC

Es un programa de libre distribución y código abierto que accede al entorno de Kinect y envía los datos de las articulaciones por un puerto seleccionable. Permite elegir el modo de codificación de las coordenadas XYZ (las tres juntas, separadas, o todo el esqueleto en un solo mensaje). Utiliza el protocolo Open Sound Control, que no necesita establecimiento de conexión. El programa se ha ejecutado sobre un entorno VisualStudio, codificado en lenguaje C#.

- Addon

Es un fichero programado en Python, creado en semestres anteriores en el CITSEM, usado como plug-in para Blender. Interpreta los datos que envía KinectOSC y crea un esqueleto en Blender que responde a los movimientos del jugador. Este esqueleto puede ser de rotación, de posición, mixto de ambos o de amplificación de movimientos.

6. COMPETENCIAS Y HABILIDADES ADQUIRIDAS CON LAS PRÁCTICAS

Estas prácticas me han permitido un conocimiento en mayor profundidad del programa Blender, el cual nos enseñaron en la asignatura **Síntesis y Animación de Imágenes**. Esta asignatura se basa en la explicación general del programa, centrándose en el modelado de objetos. El trabajo final de esa asignatura era la creación de un videojuego, empleando el motor de juegos (Blender Game Engine). Gracias a las prácticas, además de haber aprendido nuevas funciones de este motor de juegos que desconocía, he aprendido a programar scripts en Python para Blender, que permiten acceder a objetos y propiedades, además de otras muchas funciones inaccesibles si se usasen los bloques lógicos.

He aprendido a programar en lenguaje Python prácticamente desde cero. Me he tenido que habituar a su forma de programar funciones y clases, a cómo se puede acceder a las propiedades de otros objetos, de otras escenas. El aprender a programar sin tener conocimiento previo de ese lenguaje de programación me ha ayudado haber cursado las asignaturas **Programación I** y **Programación II** en las que me enseñaron a programar en C y en Java.

7. CONCLUSIONES

Este proyecto puede llegar a ser un buen método de rehabilitación que sea también entretenido y ameno. Es un proyecto ambicioso pero realizable. En estas prácticas se han realizado varios ejercicios que pueden formar parte de este gran proyecto global. La idea principal era poder realizar juegos basándose única y exclusivamente en la detección de los movimientos del usuario; y se ha conseguido.

Al principio fue complicado adaptarse a realizar la lógica del juego empleando los datos captados por la Kinect, en vez de los clásicos teclado y ratón. Ha servido de gran utilidad el addon creado en semestres anteriores, que permite directamente crear un esqueleto que recibe los datos de la cámara. Además, gracias a este addon, se pueden añadir dos objetos que se han utilizado en gran medida para los ejercicios que son las cuerdas y el acelerómetro. Las cuerdas permiten verificar si ciertos objetos están cerca unos de otros, y el acelerómetro devuelve en cada momento la velocidad y aceleración del objeto al que está emparentado, tanto global como en cada uno de los ejes. Muy útil para la creación de la lógica de los juegos.

Para la detección del movimiento del usuario, se ha utilizado la Kinect de Microsoft para la consola Xbox 360 (siendo esta cámara compatible con el ordenador). Ha cumplido con su misión de localización y movimiento del usuario pero la detección en las manos y los pies es muy deficiente, puesto que es muy poco estable y varía continuamente de valor.

Para mejorar este proyecto en un futuro sería muy recomendable actualizarse a la Kinect para la Xbox One, que detecta mayor número de puntos en cada parte del cuerpo incluyendo varios puntos de detección en la cara, lo que podrá ser muy interesante para el desarrollo del juego. Además tiene mayor estabilidad el esqueleto captado de la posición del usuario.

8. DIARIO DE PRÁCTICAS,

Semana 1: del 8 al 14 de febrero. Inicio de las prácticas. Reunión con la tutora y el grupo de trabajo para explicación del proyecto

Semana 2: del 15 al 21 de febrero. Reunión con grupo de trabajo para la organización del proyecto

Semana 3: del 22 al 28 de febrero. Inicio de trabajo con Blender y Kinect.

Semana 4: del 29 de febrero al 6 de marzo. Estudio del estado del arte y primeras pruebas.

Semana 5: del 7 al 13 de marzo. Reunión con tutora y grupo de trabajo para visualización de los avances realizados.

Semana 6: del 14 al 20 de marzo. Reunión con tutora para evaluación de la lista de ejercicios

Semana 7: del 21 al 27 de marzo. Inicio del ejercicio con maza

Semana 8: del 28 de marzo al 3 de abril. Inicio de ejercicio de escalera

Semana 9: del 4 al 10 de abril. Reunión con tutora y grupo de trabajo para visualización de los avances realizados.

Semana 10: del 11 al 17 de abril. Inclusión de acelerómetro en el ejercicio con la maza

Semana 11: del 18 al 24 de abril. Inicio de ejercicio de la barca

Semana 12: del 25 de abril al 1 de mayo. Reunión con tutora y grupo de trabajo para visualización de los avances realizados.

Semana 13: del 2 al 8 de mayo. Mejoras en los tres ejercicios

Semana 14: del 9 al 15 de mayo. Reunión con tutora y grupo de trabajo para visualización de los avances realizados.

Semana 15: del 16 al 22 de mayo. Programación de scripts para gestionar las escenas

Semana 16: del 23 al 29 de mayo. Reunión con fisioterapeuta que evalúa la funcionalidad de los ejercicios

Semana 17: del 30 de mayo al 5 de junio. Inclusión de marcadores

Semana 18: del 6 al 12 de junio. Pruebas con personas con movilidad reducida

Semana 19: del 13 al 19 de junio. Pruebas con conocidos

Semana 20: del 20 al 26 de junio. Mejoras finales del proyecto

Semana 21: del 27 al 30 de junio. Fin de las prácticas

9. DESCRIPCIÓN DE LOS MINIJUEGOS REALIZADOS

- Escaleras

En este juego se comprueba el funcionamiento de cómo la Kinect copia los movimientos del usuario, y cómo poder emplearlo para lógica del juego. En realidad no es uno de los minijuegos que se tiene pensado para el juego final, pero sirve para entender cómo usar la lógica de Blender, y se podría usar como escalada de algún precipicio.

El objetivo es que el personaje suba hasta arriba la escalera de manos en la que se encuentra.

Hay dos avatares: el de las escaleras y el encargado de las cuerdas. Los dos copian los movimientos del usuario pero, el de las cuerdas, tiene asociado a sus dos brazos unas "cuerdas", que son sensores de proximidad que a la que se acercan a una determinada distancia del final de cada cuerda, activan una propiedad y mandan un mensaje de control a todos los objetos. Esto sirve para saber si el brazo está levantado o no. En un futuro convendría unificar los dos avatares en uno solo.

Al avatar de la escalera se le han puesto dos propiedades para saber el estado de los dos brazos: en una se controla si el brazo izquierdo está levantado (valor True) o si no lo está (valor False); en la otra lo mismo para el brazo derecho.

Al iniciar el juego, el personaje está con el brazo derecho agarrado a la escalera mientras que el brazo izquierdo está copiando los movimientos del usuario. Al levantar el usuario el brazo izquierdo, el personaje deja de copiar sus movimientos y salta una animación en la que el personaje sube un peldaño y el brazo izquierdo queda fijo a la escalera, y el personaje ahora copia el movimiento del brazo derecho del usuario. Y así sucesivamente. La lógica que se ha empleado para esto es que sólo se produce todo este cambio si en el avatar la propiedad de brazo izquierdo está con valor verdadero (brazo izdo levantado) y la del brazo derecho está con valor falso (brazo dcho no levantado).

Se ha comprobado que, usando esta lógica, hay ocasiones en las que si el usuario tiene los dos brazos levantados se producen resultados no esperados.

El juego está compuesto por tres capas: la principal, la del marcador y la final. En la principal es donde se desarrolla la acción. La del marcador indica en todo momento la distancia que le queda al usuario para alcanzar la cima; para ello emplea el script programado en Python llamado "dist.py". La capa final muestra cuántos segundos se ha tardado en llegar a la cima.

Al iniciar el juego en la capa principal, se inicia como capa superpuesta la capa del marcador; y se inicia como capa de fondo (background) la capa del final, para que inicie el contador. A la que la distancia entre el usuario y la cima es cero, se suspende la capa final (para que pare el contador) y se eliminan las otras dos capas.

Empleando en Python la sentencia: y = "%.2f" % (x)

se guarda en ‘y’ (que es de tipo String) el valor de x redondeándolo a dos decimales. Esto se ha empleado puesto que si no, tanto el contador como la distancia a meta se expresaba con excesivos decimales.

El juego está en el fichero escaleraAmp.blend, en el que aparece comentado el código Python correspondiente para evaluar la distancia a la cima (“dist.py”).

Al adaptar este minijuego para usarlo con las gafas de realidad virtual Oculus Rift, hay detalles a considerar. La lógica del juego inicialmente se realizó para que se viera desde una cámara por detrás del usuario, pero, como al usar las Oculus Rift la cámara se sitúa en la posición de la cabeza del personaje habría que realizar cambios.

Para la lógica del juego se hizo que el avatar solo subiera al subir el brazo derecho (por facilidad al superponer las animaciones). Usando la vista desde detrás del personaje no hay problema pero al usar las gafas de realidad virtual queda raro que solo se vea subir cuando sube el brazo derecho y no cuando sube el izquierdo.

Otro tema que se plantea es que al poner la capa final como background, utilizando la vista desde atrás no hay problema. Pero al usar las Oculus Rift, como puedes mirar en todas direcciones, si no hay un techo que lo recubra todo, se verá el texto correspondiente a la capa final. Una posible solución, que es la que se empleó al realizar las pruebas, es dejar los objetos de la capa final como ocultos y, al llegar a la cima, antes de suspender la capa final, poner en el script que se puedan ver esos objetos.

El juego para las Oculus Rift está en el fichero escaleraAmpOculus.blend

- Barca

En este ejercicio el personaje está en una barca en el mar y, mediante el movimiento de sus brazos, tiene que remar hasta una meta. El movimiento de los brazos del personaje es igual al del usuario gracias al uso del Kinect.

El personaje tiene que remar entre unas boyas hasta llegar a la meta. En este ejercicio, la barca se desplaza siempre en línea recta pero, para el proyecto general, se planteó que pudiera ir girando a voluntad del usuario.

Para saber en todo momento la posición de los brazos se han emparentado al avatar del personaje cuatro “cuerdas”, dos por cada brazo. Hay que recordar que las cuerdas en realidad son sensores de proximidad. El inicio de las cuerdas está en las manos (dos en cada mano). El final de las cuerdas está uno adelante y atrás (en cada brazo), de tal forma que en todo momento las cuatro cuerdas estarán mandando un mensaje a todos los objetos del juego de si el brazo está cerca o lejos de ellas.

En la lógica del juego, en el avatar, se han creado cinco propiedades, cuatro de ellas relacionadas con la posición de los brazos. Estas cuatro propiedades son: LF (Left Forward), RF (Right Forward), LB (Left Back), RB (Right Back). Según el mensaje que reciban de las cuerdas, estas propiedades se pondrán a True o a False.

La otra propiedad es la llamada “adelante”, que se pone a verdadero cuando los dos brazos están hacia delante (LF y RF con valor True). Al echar los brazos hacia atrás (LB y RB a True) y estar la propiedad “adelante” a con valor True, se desplazará la barca.

Se ha observado que en ocasiones, al desplazarse varias veces, cambia el orden de la lógica y la barca empieza a desplazarse cada vez que los brazos se mueven hacia adelante. Habría que averiguar por qué ocurre esto. También se ha observado que solo se puede desplazar la barca únicamente cuando el anterior desplazamiento ha cesado. Convendría acortar el final de la animación del desplazamiento.

Tras realizar las pruebas con niños se ha comprobado que quizás es un ejercicio demasiado largo y repetitivo. Bastaría con añadir elementos de decoración y/o acortar la distancia a la meta.

El juego está formado por tres capas: “Scene” que es la capa principal, “Marcador” donde se muestra la distancia a la meta y “Final” donde se muestra cuánto tiempo se ha tardado en llegar a la meta y la velocidad media empleada.

Se ha programado un script muy semejante al utilizado para la escalera, llamado “distancia.py” en el que en todo momento se evalúa la distancia la distancia a la meta y se muestra en el juego mediante texto. Como ya se explicó en el apartado de la escalera, para evitar que haya excesivo número de decimales, se ha redondeado empleando la sentencia: y = "%.2f" % (x). Para ello y tiene que ser un string. El objeto de texto “Crono” de la escena “Final” está siempre oculto y, a la que la barca alcanza la meta, se redondea con dos decimales el valor de su propiedad “Text” (Timer) a la propiedad “Text” (String) del objeto de texto “T_Crono”.

Es un poco confuso que las propiedades de objetos de texto se tengan que llamar todas iguales, pero Blender no permite que se cambie su nombre.

El juego está en el fichero remarAmp.blend, en el que se encuentra comentado el código Python que evalúa en cada momento la distancia a la meta (distancia.py).

Para usar este juego con las gafas de realidad virtual, nuevamente hay que ocultar todos los objetos de la escena “Final” y únicamente hacerlos visibles cuando se haya alcanzado la meta.

El juego para las Oculus Rift está en el fichero remarAmpOculus.blend

- Guacamole

En este ejercicio el personaje del juego tiene una maza en su brazo derecho. El personaje copia los movimientos del usuario, mediante lo captado por la Kinect. Está configurado para que en este juego se copien los movimientos de los brazos. De cuatro agujeros que hay en el suelo, van a ir saliendo topos que tienen que ser golpeados por el usuario. En total van a aparecer veinte. El objetivo es golpear al mayor número de topos.

Asociada a la maza hay un acelerómetro que mide la aceleración y velocidad a la que se mueve la maza. Si la maza supera una determinada velocidad envía un mensaje al resto de objetos del juego de que se mueve rápido. De lo contrario informa que se mueve lento.

Cada topo tiene cuatro propiedades: una propiedad booleana que evalúa si la maza/martillo se mueve rápido ("martilloRapido"), una propiedad float que se usará como contador de cuánto le queda para subir ("contadorSubir"), una propiedad float que se usará como contador de cuánto le queda para subir ("contadorAguantaArriba") y una propiedad booleana que evalúa si el topo ya ha sido golpeado.

En cada segundo se decrementa en una unidad el valor de las propiedades "contadorSubir" y "contadorAguantaArriba" (aunque para se actualice más rápido, cada media décima de segundo se resta 0.05 del valor de esas propiedades).

Al iniciar el juego se da un valor aleatorio a la propiedad "contadorSubir". El topo subirá cuando esta propiedad tenga un valor entre -1 y 0, y además, se da un valor aleatorio a la propiedad "contadorAguantaArriba" y se informa de que ha salido un nuevo topo.

Si el topo es golpeado cuando la maza se mueve rápido y no había sido golpeado antes, entonces la propiedad "golpeado" se pone a True, se avisa que un topo ha sido golpeado mediante un mensaje y salta la animación de bajar junto con la de topo golpeado.

Si el topo no ha sido golpeado y la propiedad "contadorAguantaArriba" llega a 0, entonces únicamente salta la animación de bajar.

Cuando la propiedad "contadorAguantaArriba" llega a un valor entre -6 y -5, se le da un valor aleatorio a la propiedad "contadorSubir" y la propiedad golpeado se le da un valor de false.

Durante el juego hay una escena superpuesta que indica cuántos topos lleva golpeados (escena "Texto Contador").

El juego se acaba cuando hayan salido ya 20 topos. Cuando esto ocurre sale la escena final en la que se indican cuántos topos han sido golpeados de los veinte que han salido.

Este juego no utiliza controlador Python. El control de la lógica entre escenas se ha utilizado en la lógica del objeto suelo (en realidad daba igual donde usarlo).

En la escena "Texto Contador" se controla la lógica de cuántos topos han salido ya y cuándo hay que acabar el juego.

Durante las prácticas se ha observado que hay ocasiones en las que al golpear a dos topos a la vez, solo lo cuenta como uno.

El juego se encuentra en el fichero guacamoleAmp.blend.

El juego para las Oculus Rift está en el fichero guacamoleAmpOculus.blend