UNIVERSIDAD DE GRANADA

GRADO EN INGENIERÍA INFORMÁTICA

SISTEMAS GRÁFICOS

CURSO 2022 / 2023

Memoria Práctica 2

Harry Potter y el Torneo de los Magos

Reyes García, Teresa Fernanda
Velázquez Ortuño, Diego

30 de mayo, 2023

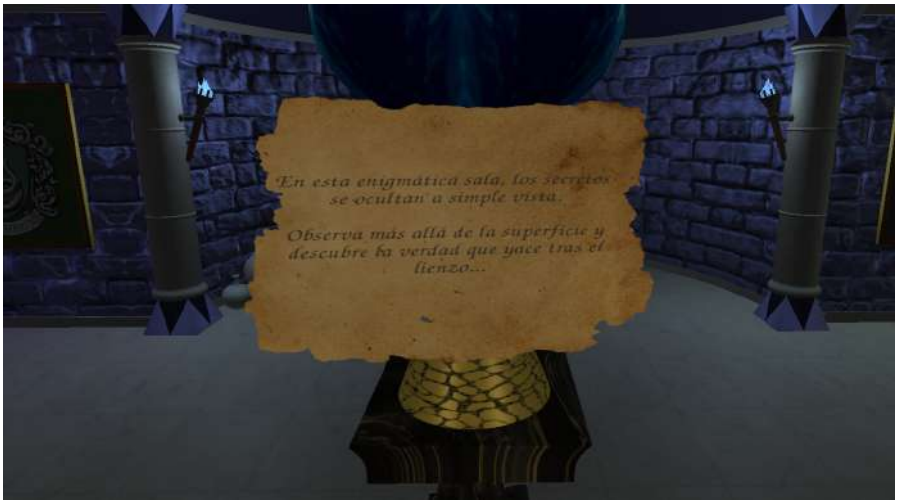
ÍNDICE

DESCRIPCIÓN GENERAL.

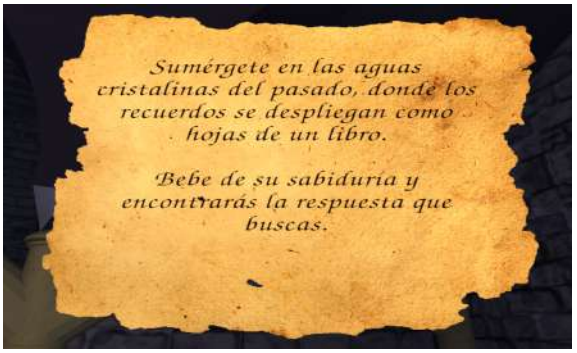
El juego de escape room tematizado en Harry Potter te lleva a través de una emocionante aventura en el mundo mágico. Te encuentras en un lugar oscuro y misterioso que parece ser un aula de pociones de Hogwarts (la Escuela de Magia y Hechicería). La habitación está decorada con elementos del universo de Harry Potter, como retratos de las casas de Hogwarts, pociones, un gran caldero y otros objetos mágicos.

El **objetivo** del juego es completar una serie de pruebas que te llevarán a través de la habitación que habrá que superar para encontrar la salida.

- El **primer puzzle** que hay que completar es clicar encima del modelo jerárquico, el maniquí, el cuál está caminando por una pasarela. Tras seleccionarlo, este automáticamente se recolocará posicionando su varita y disparando un hechizo a una esfera que flota encima de uno de los objetos de la sala. Esto producirá una gran fuente de luz verde, que llegará al maniquí y le matará. Además, aparecerá un papel enfrente del objeto golpeado por el hechizo, con la siguiente



- El **segundo puzzle** consta de encontrar detrás de uno de los diversos cuadros, colgados en las paredes donde se sitúan las estanterías, un papel con la siguiente pista del escape room. Al clicar en ellos, los cuadros se abren y se enciende una luz focal para que se pueda ver el contenido que hay detrás.



- La **tercera pista** consta de ir al pensadero, un cáliz lleno de agua, y beber su contenido clicando repetidas veces sobre el agua. Cuando ya no queda líquido, se puede ver en su interior la llave que abre la puerta de salida de la sala,

ANOTACIONES

- Hemos ido referenciando líneas de código a lo largo de la memoria para especificar en qué partes de nuestro código se realizan ciertas tareas.

- Recomendamos que al iniciar el juego, se espere uno o dos minutos antes de comenzar a realizar los puzzles para que carguen correctamente todos los objetos y animaciones

REQUISITOS.

1. Geometría.

Para crear el escape room, nos hemos inspirado en uno de los juegos oficiales basados en la saga de Harry Potter. Tomamos prestada la estructura de la habitación principal, incluyendo las bóvedas y columnas, para recrearla en nuestro diseño. Además, nos inspiramos en la disposición de estanterías a ambos lados de la sala, donde colocamos varios objetos como libros y pociones para agregar detalle y ambientación al entorno.

Finalmente nuestra sala ha quedado de la siguiente manera:

1.1 – Geometrías basadas en extrusión.

- **Extrusión propia** – hemos hecho uso de shapes para definir la forma de ciertas geometrías a los que le hemos aplicado la extrusión con sus características como el redondeado de bordes para objetos como el marco de la puerta o la segunda pared en las paredes laterales:

- **Barrido** – hemos aplicado extrusión por barrido o extrusión siguiendo un camino (path) que se ha obtenido de un shape en la que se definía la forma de arco, para los arcos que sostienen la segunda pared:

1.2 – Geometrías basadas en revolución.

En nuestra escena podemos encontrar geometrías por revolución que parten del uso de shapes definidos en objetos como antorchas, calderos y columnas entre otros:

1.3 – Geometrías basadas en booleanos.

- **Unión**: se han creado objetos mediante unión para poder tratarlos como un único objeto o para poder aplicar un único material a dicha geometría. De entre todos los objetos de unión, algunos ejemplos son partes de la estructura como cada pared, el suelo o las columnas, u objetos como el “pensadero” o la puerta:

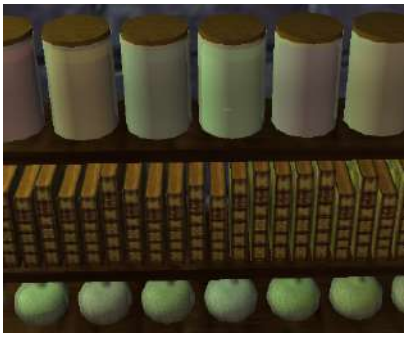
(Las columnas han sido necesarias crearlas por unión de un cilindro un objeto por revolución a partir de shape. Esto se debe a que la textura no se aplicaba bien al shape, por lo que se ha asignado al cilindro y luego, éste se ha unido con el shape por revolución, haciendo que la textura se reparta de forma uniforme).

- **Disyunción**: hemos usado la disyunción o resta para generar estructuras más complejas como las paredes circulares (eliminando un cilindro a otro), el techo (eliminando a un cubo de gran tamaño las esferas, cilindros y figuras por extrusión en forma de pico para las bóvedas), el hueco en la pared para la puerta, el hueco en la pared, etc:


1.4 – Geometrías pre-implementadas.

En la escena se han usado todo tipo de geometrías básicas para formar otras nuevas o para añadirlas directamente en la escena. Algunas son cubos, cilindros, esferas, prismas, toros, cápsulas, planos:





1.5 – Objetos cargados.

Hemos implementado modelos más complejos cargados a los que le hemos aplicado materiales propios. Estos han sido, la varita, la figura de la mesa, la taza de la mesa (se le ha aplicado el material que ella misma traía), las velas de la mesa, el pedestal del objeto mágico y el atril del libro:



2. Animación

2.1 – Puerta.

Hemos implementado una puerta de salida que se abrirá cuando se completen todos los puzzles. Para interactuar con la puerta y abrirla, hemos diseñado una mecánica sencilla que requiere hacer clic con el ratón en el pomo de la puerta y no en cualquier otra parte de ella (`1.230 – 248, H_estructura.js`):

2.2 – Modelo jerárquico.

El **modelo jerárquico** que hemos elegido para el proyecto es un maniquí de lucha articulado que mueve los brazos, cintura y las ruedas que tiene como pies (`1.21 – 221, maniqui.js`). Cuenta con varias animaciones:

- La animación inicial donde el modelo se encuentra en un movimiento continuo, moviéndose sobre la plataforma donde está colocado hacia delante y hacia atrás. El movimiento de las ruedas acompaña a la traslación del modelo, siendo un movimiento lo más realista posible. (**`1.303 – 372, maniqui.js`**).

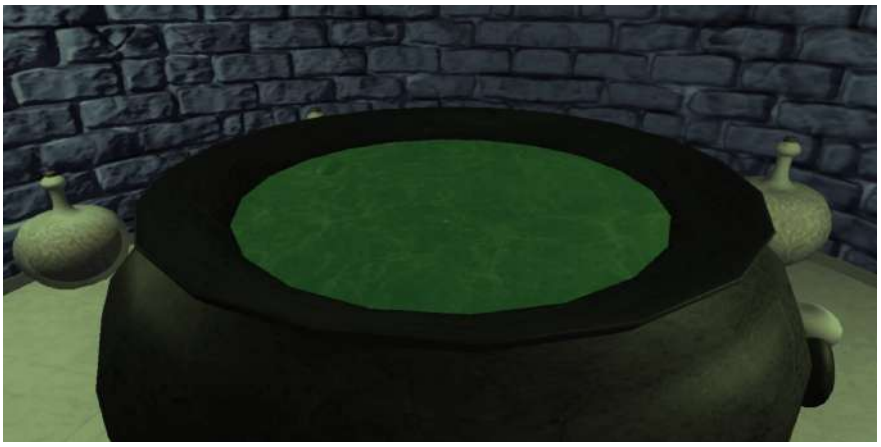
- La siguiente animación se realiza al clicar encima del modelo, donde este (sin importar en qué parte de la pasarela se encuentre) se para y se recoloca posicionando su varita y levantando el brazo para lanzar un hechizo. (**`1.376 – 427, maniqui.js`**).

- La penúltima animación es cuando el modelo lanza un hechizo a uno de los objetos que hay colocados enfrente de la pasarela, una especie de esfera que al comenzar se encuentra en negro y, tras el hechizo, comienza a reproducir un video. (**`1.432 – 505, maniqui.js`**):

- La última animación se trata de que una vez lanzado el hechizo, el maniquí se ve afectado por su magia, pierde el control propio (empieza a girar muy rápido) y finalmente muere dejando caer sus brazos y cabeza (**`1.509 – 564, maniqui.js`**):

2.3 – Caldero.

Hemos incorporado una animación infinita en el caldero para simular el efecto de que el líquido esté hirviendo. Esta animación consiste en burbujas que suben y bajan constantemente dentro del caldero, mediante recorridos o splines (**`1.405 – 616, decoracion.js`**).



2.4 – Cuadros.

Para los cuadros se usa una animación parecida a la de la puerta, una apertura por engranajes, con la diferencia de que en cualquier momento de la apertura se puede volver a seleccionar y el cuadro se volverá a cerrar partiendo del punto en el que estaba (**`1.862 – 887, MyScene.js`**):

2.5 – Objeto mágico.

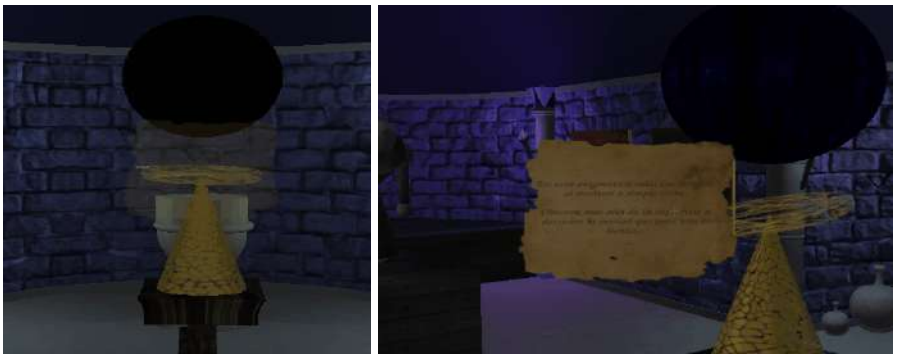
El objeto mágico consta de 3 animaciones:

- La rotación en la X y en la Y que simula una onda (`1.1074 – 1104, decoracion.js`):
- La animación de cambio de valores de intensidad de la luz y de la emisión de la esfera (`1.479 – 492, maniqui.js`):

2.6 – Papel pista.

Uno de los dos papeles de pista tiene dos animaciones más:

- Cambio de transparencia desde transparente completamente hasta ser opaco (`1.472 – 475, maniqui.js`):
- Rotación en todas direcciones actualizando con *lookAt* indicando el punto donde está la cámara como objetivo (`1.807, MyScene.js`):



2.7 – Libros especiales.

Una de las animaciones de decoración es la del libro mágico flotando, que es un simple cambio de posición en la Y (`1.373 – 408, MyScene.js`):

3. Interacción

3.1 – Movimiento

El jugador podrá moverse por la sala haciendo uso de las siguientes teclas:

- Tecla **W** – avanzar hacia delante.
- Tecla **S** – avanzar hacia atrás.
- Tecla **A** – avanzar hacia la izquierda
- Tecla **D** – avanzar hacia la derecha.

Adicionalmente, hemos incorporado la función de agacharse al presionar la tecla Shift, lo cual baja la cámara sutilmente y permite al jugador moverse a una velocidad más lenta. Esto se consigue modificando la altura de la cámara si está pulsada la tecla o volviendo a su altura original si se suelta (**`1.760, MyScene.js`**).

En el movimiento también está implementada una funcionalidad de comprobación de si el movimiento está bloqueado o no (**`1.748,772, MyScene.js`**)

La implementación actual permite utilizar dos teclas simultáneamente para moverse en diagonal, por ejemplo, W + A para avanzar diagonalmente hacia la derecha. Para esto, en una única actualización se comprueban las 4 opciones posibles de movimiento, y no sólo actúa la primera que se activa (se usa if..if..if en vez de if..else if..else) (**`1.785 – 799, MyScene.js`**).

3.1 – Colisiones

Para poder moverse por la escena, antes de llamar a las funciones del controlador de la cámara, se comprobará si se puede avanzar en la dirección deseada. Para ello primero obtendremos la dirección horizontal (*y=0*) en la que apunta la cámara y, si quiere avanzar al frente, se comprueba la colisión con la dirección sin modificar, pero si quiere avanzar hacia atrás, derecha o izquierda, el vector dirección se niega o se hace el producto vectorial con el vector vertical según la dirección que se quiera (**`1.776 – 783, MyScene.js`**).

Para comprobar las colisiones hemos hecho uso del *RayCasting*, generando dos rayos distintos:

- Un rayo **horizontal** en la dirección a la que apunta la cámara con la altura de la cámara (**`1.833, MyScene.js`**).

- Un rayo **vertical** hacia abajo que comienza en un punto cuya *y* es la altura de la cámara y cuyos *x* y *z* se calculan sumándole a la posición de la cámara el vector de su dirección normalizado y multiplicado por un espacio determinado, lo que hará que se detecte colisión con toda la vertical del “cuerpo” (**`1.845, MyScene.js`**).

3.2 – Selección de objetos

Para comprobar si un objeto es seleccionado o no, partiremos una lista en la que tendremos, de todos los objetos de la escena, sólo aquellos que son seleccionables. Al hacer clic con el rato (*onMouseDown*), usaremos *RayCasting* con un rayo desde la posición de la cámara pasando por el centro de la pantalla (*0,0*) y se obtendrán los objetos con los que ha intersectado con la restricción de que se evaluarán sólo los que estén el el vector de seleccionables (**`l.926, MyScene.js`**).

Si ha habido un objeto intersectado, se comprueba, por su nombre, cúal de los seleccionables ha sido, y se actúa en consecuencia según lo que debe hacer cada uno. Algunos tienen restricciones según las cuales, un objeto no puede ser seleccionado hasta que no se cumpla una condición (**`l.946 – 978, MyScene.js`**).

3.2.1 – Apuntado de objetos

Adicionalmente, cada objeto seleccionable está compuesto por su estructura (geometrías, materiales, animaciones…) y por una segunda geometría que resulta de una clonación de la geometría principal (escalada) y un material con las caras al revés y de color neutro (esta será la **silueta** o **contorno** del objeto) que inicialmente no será visible.

Esto se usará para poder saber si el objeto al que estamos apuntando es seleccionable en ese momento o no. Para ello, en cada actualización se lanza un rayo que pasa por el centro y que intersecta sólo con los objetos seleccionables y se comprueba si, en ese momento, dicho objeto se puede seleccionar, siendo un funcionamiento parecido al anterior. La diferencia con el anterior radica en la necesidad de comprobar en todo momento a dónde apunta la cámara (por lo que se hace en cada actualización) y no sólo al clicar. Finalmente, si el objeto se puede seleccionar, se accede a su silueta mediante el nombre (todas las siluetas de los objetos tienen el mismo nombre y están al mismo nivel en cuanto nodos) y su visibilidad se activa (**`l.686 – 714, MyScene.js`**).

Si la implementación se quedara así, surgiría el problema de que al apartar el puntero de la figura, su contorno se seguiría viendo, pues no se ha vuelto a poner a false su visibilidad. Para arreglar esto, añadimos una variable global que referencia al objeto apuntado anteriormente y este se asigna en el momento en el que un objeto es seleccionable y su silueta es visible.

En siguientes actualizaciones (en las que ya no se apuntará al objeto), el número de objetos seleccionables intersectados será cero y se comprobará si el anterior apuntado es distinto a null y, si es así, la visibilidad de la silueta de éste será false, y el anterior apuntado se pondrá a null, pues ya no hay ningún contorno activo (**`l.722, MyScene.js`**).

3.2.2 – Objetos superpuestos

Inicialmente, surgió el problema de que, tanto en la selección como en el apuntado, cuando seleccionábamos un objeto seleccionable a través de otro que no lo era (p.e. una columna), se podía seleccionar de igual manera ya que la intersección del rayo sólo se comprobaba con los objetos seleccionables:

Para solucionar esto, creamos un segundo rayo con la misma dirección que el rayo de selección o de apuntado respectivamente, pero que intersecta con todos los objetos de la escena y se comprueba si la distancia del objeto **NO** seleccionable más cercano es la misma o mayor que la del objeto seleccionable más cercano.

Si esta distancia es menor significa que el objeto al que se apunta no es uno seleccionable o no está por detrás del mismo, lo que significa que está entre el objeto seleccionable y la cámara, por lo que no se podrá interactuar con él (`1.690,934, MyScene.js`).

4. Materiales

4.1 – Materiales basados en un color

En la creación de nuestro escape room, hemos utilizado una variedad de materiales para dar vida a nuestros objetos y superficies. Entre los materiales que hemos utilizado se encuentran los siguientes, cada uno con sus propias características y efectos visuales.

- **MeshLambertMaterial** – hemos usado materiales del modelo de Lambert con componentes ambiental, difusa y emisiva para la mayoría de los objetos de la sala debido a que nos ha permitido un control de variables como color, opacidad o sombreado, creando así diferentes efectos en los objetos.

  Gracias a esto hemos creado objetos como las pociones y frascos, con las variables `transparent: true, opacity: 0.8` (con textura para transparencia), que nos han permitido simular cristal:

- **MeshPhongMaterial** – hemos usado este material para objetos que requieren un brillo especular más pronunciado y una mayor reflectancia, como podría ser el suelo de mármol, objetos brillantes como el objeto mágico.

- **MeshBasicMaterial** – hemos usado materiales básicos para las siluetas de los objetos seleccionables, para los papeles de las pistas, así como para el plano usado para el paisaje, en el exterior de la sala. Esto lo hemos hecho ya que a este tipo de material no le afectan las diferentes luces que tenemos colocadas en la escena y se van a ver siempre iluminados.



Además, hemos hecho uso de otras clases como…

- **MeshToonMaterial** (para el objeto mágico).
- **MeshStandardMaterial** (para el video de la esfera).
- **MeshMatcapMaterial** (para ir modelando objetos y visualizarlos).

4.2 – Materiales basados en texturas para el canal difuso

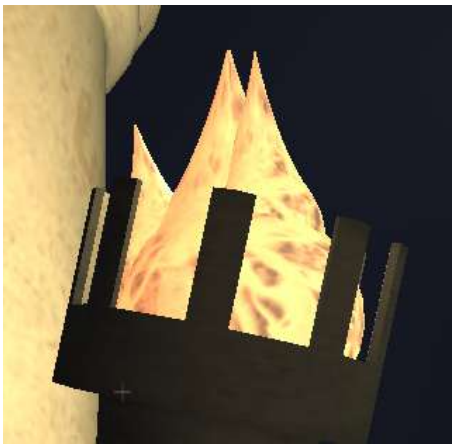
Hemos utilizado materiales basados en texturas para el canal difuso con el fin de agregar color, textura y detalles visuales a nuestros objetos 3D. Al aplicar estas texturas en el canal difuso de nuestros materiales, hemos logrado crear superficies más realistas. Algunos ejemplos son los siguientes:


4.3 – Materiales con una textura en el canal de relieve

Hemos utilizado materiales con texturas en el canal de relieve para agregar profundidad y detalles a nuestros objetos 3D. El uso de estas texturas en el canal de relieve nos ha permitido simular la apariencia de superficies rugosas, en relieve o con relieves sutiles, como por ejemplo, las paredes. En este caso, hemos usado mapas de normales.

4.4 – Materiales emisivos.

Algunos objetos como la esfera de vídeo, las antorchas o la taza de bebida mágica de la mesa tienen cierta emisión propia de luz, que se consigue con el parámetro `emissive` y `emissiveMap` (para las texturas que emiten luz como la de la antorcha o taza), además de su intensidad:



4.5 – Configuración de las texturas

4.4.1 – Repeticiones y transformaciones.

En el proceso de configuración de nuestras texturas, hemos aplicado diversos ajustes para lograr efectos visuales específicos y mejorar la calidad visual de nuestros objetos. Entre las

configuraciones utilizadas, hemos hecho uso de repeticiones como con el modo THREE.MirroredRepeatWrapping para el caso del objeto mágico (**1.1048 – 1049, decoracion.js**).

Este objeto a su vez hace uso de transparencia, activando las transparencias y aplicando el material en ambos lados de la piel de la figura (**1.1056, decoracion.js**).

```
material_alpha.alphaMap                                                    =
this.texturaLoader.load('../imgs/alphas/textura_alpha1.jpg');
material_alpha.transparent = true;
material_alpha.side = THREE.DoubleSide;
```

También repetimos texturas para las paredes, con una función que hemos creado `repeatTexture(texture, repeatX, repeatY, espejo)` (**1.24, H_estructura.js**).

4.4.2 – Entorno.

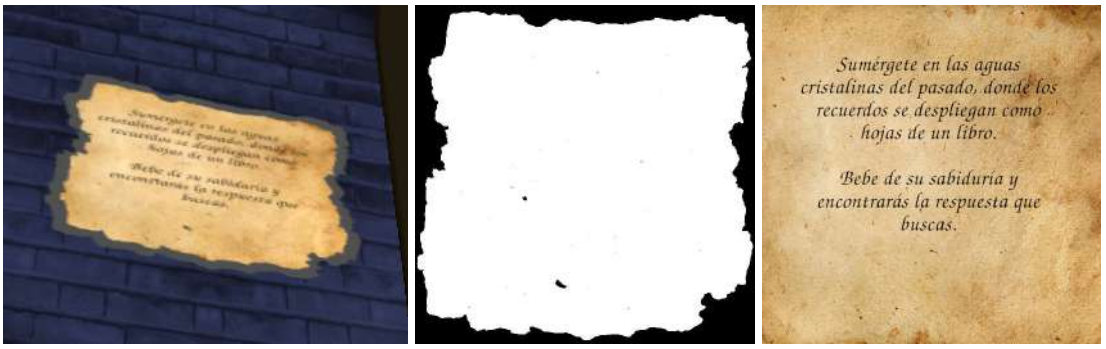
Hemos creado un objeto libro, (**1.118, decoracion.js**), que consta de diferentes texturas en cada una de sus caras, representando la portada, contraportada, lomo y páginas. Esta configuración nos permite simular un libro completo con todos sus elementos mediante el uso de texturas individuales.

4.4.3 – Usar un video como textura.

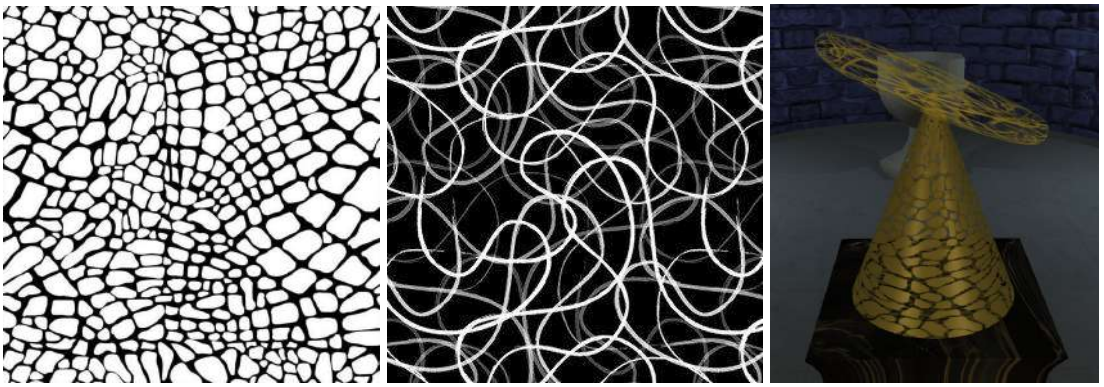
Hemos incorporado, encima del objeto mágico, una esfera que utiliza un video como textura. En lugar de utilizar una imagen estática, hemos optado por aplicar un video como textura a la superficie de la esfera para simular la magia de la sala. Cuando el maniquí lanza el hechizo, este colisiona con esta esfera y es cuando el video comienza a reproducirse (`l.1111, decoracion.js`).

4.4.4 – Texturas recortadas.

Hemos utilizado texturas recortadas para lograr efectos visuales precisos y detallados en nuestros objetos 3D. Este tipo de texturas las hemos usado para crear los papeles para dar la sensación de ser un papel antiguo, como un pergamino (`l.123, Myscene.js`).



Al igual pasa con el objeto mágico:



5. Luces

En nuestro proyecto, hemos creado antorchas para iluminar la sala, incorporando luces de varios colores a lo largo del entorno. Estas antorchas desempeñan un papel fundamental en la ambientación del juego.

Además de las antorchas, cuando los cuadros se abren en el juego, hemos incorporado una luz focal específica que ilumina el área detrás de ellos. Esta luz focal sirve como un punto de enfoque y ayuda a los jugadores a ver claramente lo que hay detrás de los cuadros, revelando detrás de uno de ellos la pista que te conduce al último puzzle.

En la escena hay una luz que también se activa (una única vez) tras una animación que es la de la esfera del vídeo cuando recibe el rayo, cuya intensidad en un principio comienza a

cero y aumenta y disminuye simulando una explosión. Además el rayo, que también tiene con textura emisiva, tiene también una luz puntual que se mueve con él:

5.1 – Sombras.

Debido a que las sombras disminuyen en gran medida el rendimiento de la escena, hemos optado por aplicar únicamente sombras al pensadero:

- El pensadero proyecta sombras.
- El agua del pensadero, el suelo y la pared norte reciben sombras.
- Las dos antorchas azules y la luz de la explosión de la esfera generan sombras.

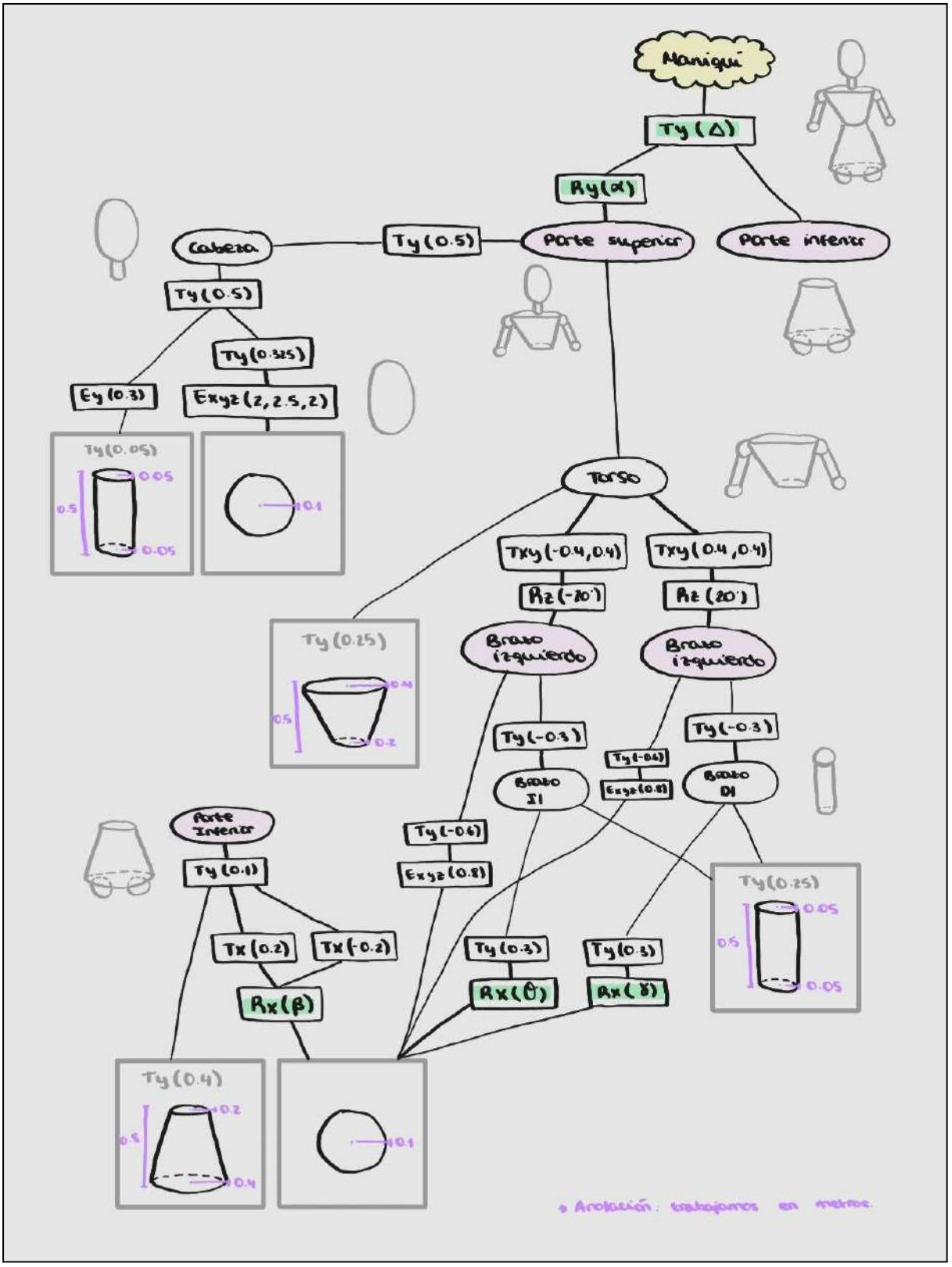
6. Cámaras

Es un juego en primera persona, la cámara principal simula la visión del personaje protagonista del juego.

En nuestra escena usamos el controlador para la cámara ***PointerLock***, que permite capturar el ratón y mover la cámara únicamente al punto que queremos mirar, sin que ésta continúe su movimiento en la dirección del ratón, por lo que necesitaremos usar las funciones *moveForward* y *moveRight*, para indicarle que avance un espacio concreto hacia al frente o hacia la derecha, o ese mismo espacio en negativo para ir hacia atrás o hacia la izquierda.

Además, de forma adicional, hacemos uso de una pequeña cruceta para marcar cual es el punto de mira o crosshair del personaje, y así facilitar acciones como la selección de objetos.

MODELO JERAŔQUICO



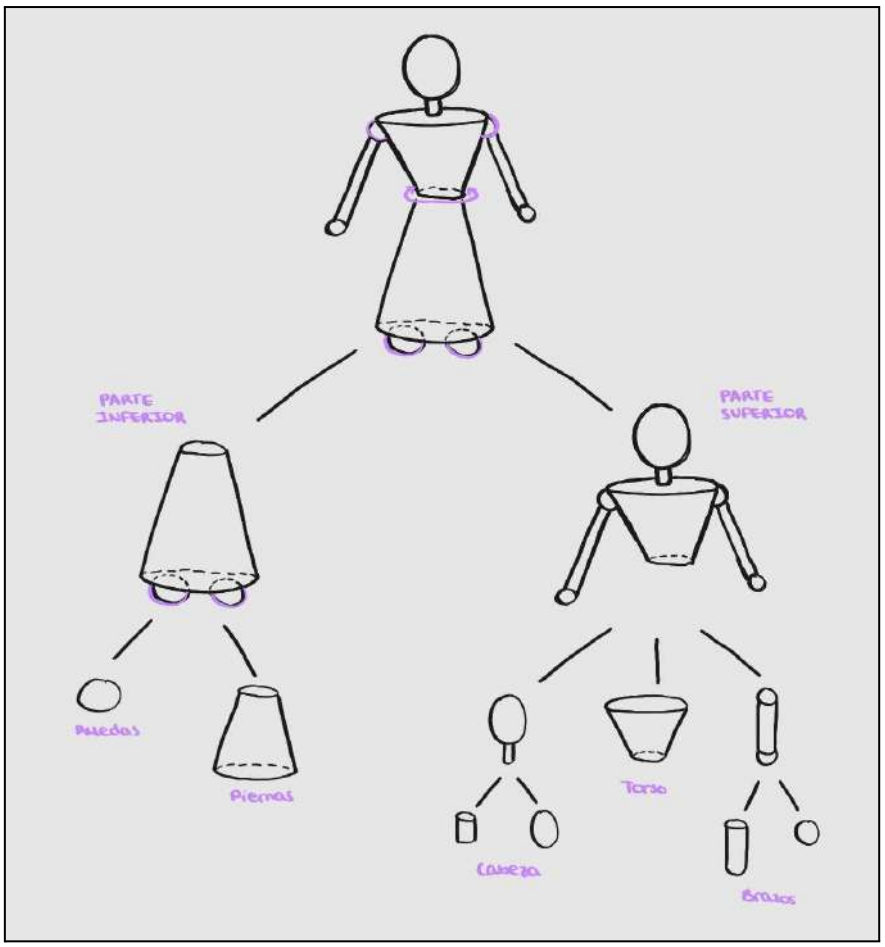

PARTE INFERIOR
PARTE SUPERIOR
Pies
Piernas
Cabeza
Torso
Brazos

DIAGRAMA DE CLASES