Técnicas Gráficos por Computadoras

AtTheEndOfTheDay

“The Perfect Element”

Integrantes:

- Crespi Gustavo
- Rodríguez Leandro
- Santoalla Gastón
- Santoalla Nahuel

Idea General

El elemento perfecto es un juego en el cual, el jugador debe hacer que una pelota entre en un cubo dimensional para ir avanzando en una serie de niveles donde deberá utilizar su ingenio y los ítems a su disposición para resolverlo.
El juego posee una apariencia 3D con una jugabilidad 2D, existe un escenario donde se encuentran la pelota, el objetivo y una serie de obstáculos característicos de cada nivel. Además la derecha hay un menú, donde el usuario podrá ver los ítems que dispone para ir resolviendo el nivel, para esto, deberá ubicarlos, rotarlos y posicionarlos de una manera conveniente para guiar a la pelota en su camino hacia el cubo.

Instrucciones de Juego

Objetivo: Que la pelota alcance el cubo dimensional en todos los niveles propuestos.

Resumen Controles:

- Click Izquierdo- selecciona un ítem de usuario y lo mueve por la pantalla, soltándolo al segundo click.
- Click Derecho- remueve un ítem del juego, devolviéndolo al menú.
- Tecla C: Pasa a la etapa de construcción
- Tecla Espacio: Pasa de Construcción a Simulación/ Pausa o reanuda el juego en la etapa de simulación
- Tecla R: Una vez completado el nivel permite reiniciarlo
- Tecla Enter: una vez completado el nivel permite avanzar al siguiente

-Dentro de la etapa de construcción con un ítem seleccionado-

- Teclas A y D: Permiten rotar los siguientes ítems en la etapa de construcción: Acelerador, Cañón (sin la base), Imán, Pared y Resorte.
- Teclas Q y E: Permiten rotar el cañón junto con su base / Permiten ensanchar o achicar las paredes hasta un limite
- Tecla W: Permite seleccionar cuál de las dos rotaciones del cañón posicionar.

NOTA: Para poder realizar una rápida inspección del juego (y de gran ayuda a la hora de armar los niveles) se agregó la función LvlHack, la que permite saltear los niveles sin haber cumplido su objetivo, pero sin recibir la satisfacción de lograrlo. Presionando F2 podrá avanzar al siguiente nivel, y presionando F1 podrá retroceder al nivel anterior.

Descripción Detallada:

Al iniciar el juego se comenzara en el nivel 1, y al ir logrando cumplir los niveles, aparecerán carteles que indicaran que el nivel fue completando, pudiendo avanzar al siguiente presionando la tecla enter.

Todo nivel se divide en dos etapas, la etapa de Construcción y la etapa de Simulación o de juego en sí mismo.

Etapa de Construcción:

Al iniciar cualquier nivel, el mismo se encontrara en esta etapa.
En esta etapa el jugador deberá colocar los ítems en la posición que desee, pero solo tendrá control en aquellos que se encuentren en el menú, aquellos que ya estén en el escenario al momento de iniciar el nivel, se consideran ítems del nivel, por lo cual el usuario no tendrá control sobre ellos. Para esto deberá hacer click izquierdo en la imagen del objeto que desee en el menú (la cual estará en una escala reducida y rotando), una vez clickeada la imagen se considera que el objeto ha sido seleccionado, cuando un objeto esta seleccionado, se lo vera de diferentes colores dependiendo su posición (ver más adelante).

Cuando el objeto este seleccionado, el usuario podrá moverlo por la pantalla moviendo el mouse y podrá rotarlo o, en algunos casos definir parte de su comportamiento durante la simulación (ver en la sección ítems, la construcción de cada uno más adelante). Una vez posicionado donde el usuario desee, deberá hacer click izquierdo para soltarlo.

Colores de un Objeto Seleccionado:

- *Verde*: El objeto se encuentra en una posición valida dentro del juego, al soltarlo (clickeando por segunda vez) se quedara fijo en esa posición hasta que el usuario vuelva a seleccionarlo.
- *Azul*: El objeto se encuentra en una posición valida dentro del menú, al soltarlo volverá a formar parte de los ítems del menú, sin ser parte del juego (tomando una vez más, su tamaño reducido y su animación de menú).
- *Rojo*: El objeto se encuentra en una posición invalida (esto se da cuando esta colisionando con otros objetos), el hacer click izquierdo no soltara el objeto, el cual seguirá seleccionado hasta que el usuario lo suelte en una posición valida.

Si el usuario deseara sacar un ítem del escenario que anteriormente hubiera colocado, podría seleccionarlo y arrastrarlo hasta el menú, o simplemente hacer click derecho sobre el ítem, lo cual automáticamente lo llevaría hasta el menú.

Una vez que el jugador coloco todos los objetos de manera estratégica para que la pelota llegue a su objetivo, puede dar por terminada la etapa de construcción presionando la tecla espacio, entrando a la etapa de simulación.

Etapa de Simulación:
Al finalizar la etapa de construcción (presionando espacio) se iniciara esta etapa.

En esta etapa el jugador deberá observar como la pelota interactúa con los objetos que el coloco, y en caso de no ganar, prestar atención al comportamiento de la misma para poder reubicar los objetos de una mejor manera.

En cualquier momento de la simulación el jugador podrá volver a la etapa de construcción (reiniciando la posición de la pelota y delos objetos con los que halla interactuado) y además podrá pausar y reanudar la simulación presionando espacio.

Es importante destacar que el nivel se completa durante esta etapa, ya que es cuando la pelota interactúa con los objetos e intenta llegar a su objetivo. En caso de lograrlo, se considera el nivel completado (y se muestra un cartel que lo indica), el usuario puede presionar la tecla enter para pasar al siguiente nivel, o la tecla R (Replay) en caso de que quiera volver a hacer el nivel (con la misma configuración de objetos con la que lo gano, pudiendo hacerle cambios si lo desea, o simplemente repetir la simulación ganadora).

Ítems del Juego:

- Acelerador: este ítem permite al jugador colocar una pista por la cual la pelota puede circular. Al circular por la pista, esta hará que la pelota gane velocidad en la dirección en donde las flechas apuntan.
  Construcción: además de posicionarse con el mouse, puede direccionarlo con las teclas A y D.

- Agujero Negro: este peligroso ítem, usado como obstáculo y como recurso del jugador atraerá a la pelota en toda dirección (así como el imán la atrae, pero en toda dirección como el repulsor), pero como su nombre lo indica, si la pelota llegara a tocarlo desaparecería del juego por lo cual el jugador deberá presionar C para volver a la etapa de construcción y reiniciar el juego.
  Construcción: este objeto solo se puede posicionar con el mouse.

- Botón: más que un ítem en sí mismo, este es un auxiliar, al presionarlo desde arriba o abajo, permite cambiar las propiedades de alguno de los ítems del escenario, el jugador deberá descubrir cuál de esos ítems es el que el botón modifica y de qué manera. Este podría hacer desaparecer algún obstáculo o incluso cambiar la gravedad.
  Construcción: Se posiciona con el mouse y se rota con las letras A y D.

- Cañón: el poderoso cañón nos permitirá disparar a la pelota en la dirección que queramos, siempre y cuando lleguemos a cargarlo. Al iniciar la simulación el cañón estará inactivo, hasta que el usuario logre que la pelota entre en el orificio del cañón, cargándolo. Una vez que el cañón este cargado rotara, abandonando su rotación inicial y se colocara el posición de disparo, una vez finalizado, disparara a la pelota con cierta potencia (puede variar en distintos cañones) hacia esa dirección. En caso de que el cañón sea parte del nivel, las posiciones (inicial y de disparo) serán propias del cañón y el usuario no podrá elegirlas, en caso de que el cañón sea un ítem del usuario este podrá tanto posicionarlo donde quiera como también establecerle las posiciones que el desee (por esto es el ítem más complejo de construir)
  Construcción: además de desplazarlo con el mouse, el cañón posee 3 tipos de rotación, la primera de ellas es la rotación absoluta, en la cual rota tanto el cañón en si como su base, es decir rota todo el objeto, para lograr esta rotación el usuario podrá presionar las teclas Q y E. Las otras dos rotaciones serán las que, dejando quieta la base, definan la posición del cañón tanto para disparar como inicial. El usuario podrá definir una posición presionando las teclas A y D y una vez agusto con la posición seleccionada presionando la tecla W podrá cambiar a la otra rotación (pudiendo modificarla con A y D también) de esta manera con W cambia de una a otra y con A y D la establece como desee. (Nota: se tomara como posición inicial la posición que tenga el cañón en el momento de presionar espacio (iniciar la simulación) y como posición de disparo la otra)

- Cubo dimensional: Un pequeño cubo (normalmente del tamaño de la pelota) al cual el jugador deberá prestar especial atención, ya que el objetivo es que la pelota lo alcance (basta con que lo colisione solamente)

- Imán: aunque muy útil y versátil, a veces difícil de emplear, el imán atraerá a la pelota con una fuerza propia (puede variar en distintos imanes) pero solo lo hará si la pelota se encuentra en su campo de atracción (direccionado por sus partes planas)
  Construcción: para poder direccionar el imán para donde el jugador desee, podrá rotarlo como las paredes presionando las teclas A y D.

- Pared: este es uno de los ítems que más aparece en el juego, normalmente se utiliza como obstáculo para evitar que el camino de la pelota sea directo, pero en ciertos niveles el usuario podrá utilizarlo como uno de sus ítems para lograr su objetivo.
  Construcción: presionando las teclas A y D se puede cambiar su rotación hacia izquierda y derecha respectivamente. Además el jugador podrá agrandar y achicar el largo de la pared con las teclas Q y E, pero hasta un límite específico para cada pared (que el jugador podrá comprobar jugando)

- Pelota: junto con el cubo dimensional, es una de las protagonistas del juego, se la vera en todos los niveles y será parte de los ítems del nivel (el usuario no podrá moverla en la etapa de construcción), al iniciar la simulación se verá afectada por la gravedad y comenzara a interactuar con los ítems que el usuario haya dispuesto en el escenario.

- Portal: este misterioso ítem permite a la pelota tele transportarse a la posición del receptor. El umbral de entrada se reconocerá por su color azul y el umbral de salida por su color naranja, pero cuidado, solo se puede usar una única vez.
  Construcción: el jugador, en caso de poseerlo entre sus ítems solo poseerá el umbral azul, el receptor (por donde sale la pelota) será fijo en el escenario. Podrá posicionar el umbral azul con el mouse, pero no rotarlo.

- Repulsor: esta pequeña pelotita roja será a veces una ayuda y otras una molestia, su efecto sobre la pelota es, como lo indica su nombre repelerla. Similar a como el imán la atrae este objeto repele a la pelota, pero sin importar la dirección en la que la pelota se encuentre.
  Construcción: este objeto solo se puede posicionar con el mouse.

- Resorte: con este ítem, el usuario podrá hacer que la pelota rebote hacia cierta dirección sin perder gran parte de su velocidad, cuando la pelota rebote en la cara superior del resorte (que será la cara de madera), la misma saldrá disparada con una velocidad mayor a la que saldría si rebotara en una pared convencional.
  Construcción: al igual que la pared, podrá posicionarlo con el mouse y rotarlo con A y con D.

Decisiones de Diseño Tomadas

Jugabilidad:

Dado que el juego proponía un ambiente de resolución de acertijos utilizando diferentes ítems, se decidió tener una jugabilidad 2D, aunque el modelo en si es 3D, durante el juego solo se maneja en el eje X, Y para las interacciones normales, haciendo más fácil para el usuario como también para su diseño.

El Juego, sus etapas, diseño general:

A la hora de realizar el juego, definimos dos etapas muy marcadas. La etapa de construcción, en la que el usuario se encarga de colocar los ítems y de diseñar lo que le parece es una solución del nivel propuesto y la etapa de simulación, la cual es en sí, la etapa más importante de la aplicación, ya que es donde se hacen todas las consideraciones físicas y donde más se sobrecarga el motor por la cantidad de cálculos realizados.
Esta etapa, por su complejidad está dividida en varias fases:

- *Animación*: en esta fase se animan los objetos no interactivos (aquellos no móviles).
- *Acción*: en esta fase se calculan todos los efectos no conservativos ya que son independientes de las restricciones impuestas por los cuerpos.
- *Colisiones*: esta fase se hace en 2 partes una verifica que las animaciones sean válidas (objetos estáticos animados estén en estado valido (ej.: el cañón al rotar)) y luego las colisiones con interactivos, que son aquellas que pueden estar sujetas a restricciones físicas.
- *Reacción*: esta fase es la más importante es donde todas las colisiones son verificadas una y otra vez hasta llegar a un estado valido o hasta un número máximo de repeticiones.
- *Simulación*: una vez cumplidas todas las restricciones necesarias se simula el movimiento de todos los objetos principalmente los interactivos.
- Por último se revisa si el estado actual de la simulación logra completar todos los objetivos del nivel, de ser así se da por completo el nivel.

Elaboración de Niveles:

A la hora de armar los niveles, en el principio del desarrollo se utilizaba una clase, la cual poseía los valores de los objetos hardcodeados y los instanciaba dependiendo el nivel en el que se encontraba. Sin embargo con el objetivo de poder hacerlos con más facilidad y sin la necesidad de recompilar el juego se decidió pasarlos a un archivo de texto para luego a través de un parser (Parser.cs), se levantan y se instancian los diferentes niveles. Luego intentando llevar esto un poco más allá, se decidió que este parser levante los niveles de un archivo .xml, seteando las properties propias de cada objeto instanciado del nivel. Para esto se utilizó reflexión (reflector), una herramienta provista por .net que permite obtener la meta información de los tipos necesaria para cargar dinámicamente las clases con los datos obtenidos desde el parser.

La elaboración del mismo consta de diferentes bloques, el primero (que contiene a todos los demás), es el bloque Level, que setea propiedades generales del nivel como la música de fondo, la intensidad y posición de la luz, el cartel de victoria, etc.

Luego de la sección Level, está la sección Goals, en la cual se instancian los objetivos puntuales del nivel (todos los niveles creados poseen el mismo objetivo).

Siguiendo con la instanciación del nivel se instancian los objetos propios del nivel que no pueden ser controlados por el usuario con sus diferentes propiedades, como el menú, las paredes limite y los obstáculos y recursos fijos, como también la pelota.

Por último se deben instanciar los ítems del usuario, que son aquellos que aparecerán ene l menú y sobre los cuales el usuario tendrá control para ir avanzando por los diferentes niveles.

Los Objetivos:

En realidad el juego consta de un solo objetivo, que la pelotita llegue a tocar el cubo dimensional, sin embargo por cómo está diseñado el mismo, podrían agregarse objetivos fácilmente, ya que cada nivel posee una lista de objetivos, y estos todos entienden los mismos mensajes, así pudiendo responder si están completos con una lógica particular. Sin embargo, el único objetivo que está realmente implementado es que se de la colisión de dos objetos (los cuales son configurados por el archivo .xml).

Los Ítems:

A la hora de diseñar los ítems, decidimos no hacer diferencia de los ítems del usuario y los ítems propios del nivel. Esta diferencia la hace el nivel a la hora de construcción, para saber sobre cuales objetos tiene el usuario y cuáles no, y a la hora de restablecer las posiciones de los mismos.

De esta manera todos los ítems (incluyendo el menú por ejemplo) heredan de la misma clase ítem. Esto nos facilitó las cosas a la hora tanto de crear nuevos ítems, ya que todo el comportamiento ya estaba en ítem y en lo único que nos teníamos que preocupar era en el comportamiento particular del mismo para redefinir esa parte y nada más.

Además los mensajes que se mandan en la simulación, pudiendo así crear la misma, son iguales a todos los ítems y cada uno actúa de una manera distinta (considerando sus partes asociadas).

Existen una serie de ítems particulares que son distintos a los demás entre ellos están los carteles de victoria, el gravitor (aplicador de gravedad), pero los más importantes son el menú y los interactivos.

- *Menú:* es único en el juego, y se encarga de capturar a los ítems del usuario y aplicarles una animación especial para que el usuario vea los ítems que tiene a su disposición. Es importante destacar que los ítems del menú, solo están en la lista del menú, y no en la lista del juego, cuando el usuario los selecciona se produce el intercambio de listas, y cuando el usuario los devuelve al menú lo mismo.

- *Interactivo*: El único ítem interactivo implementado es la pelota, que es la que se ve afectada por todos los demás, incluso por la gravedad (la cual en realidad es un ítem separado del resto cuyo único objetivo es aplicarles una fuerza a los interactivos), pero por cómo está diseñado el ejemplo agregar otro ítem interactivo no presentaría grandes dificultades, solo habría que redefinir las particularidades de ese ítem (su masa y la mesh entre otras) y se comportaría de manera similar a la pelota actual.

Las Partes:

Parte de la versatilidad que se buscó a la hora de realizar los ítems, consistía en independizar el concepto del ítem (su comportamiento en sí) del comportamiento de sus partes. Considerando partes del ítem, sus collider, y sus meshes. Dado que algunos de los ítems tienen más de una mesh y más de un collider (como el cañón), se decidió tener una lista de las partes, las cuales se suscribían a ciertos eventos (como el cambio de la escala, posición y rotación del ítem) y se iban adaptando a la vez.

Este desacoplamiento del ítem con sus partes, nos permitió por ejemplo tener partes de un ítem estáticas, mientras otras se movían. Por ejemplo el cañón permite rotar la parte superior independientemente de la base, o el portal, permite mover el umbral de entrada sin que el usuario tenga control del de salida.

Además de esta manera, podemos controlar en las diferentes etapas de simulación del juego con que objetos deseo interactuar y con cuáles no.

- *Collider*: Existen diferentes tipos, algunos siguen al objeto en todas sus transformación y otros, como el ObbTranslatedUnRotatedCollider, el cual además de estar trasladado del centro de origen del objeto no rota siempre con el mismo. Sin embargo quizás el más complejo es el HollowObbCollider, que es subclase del CompositeCollider (Un collider conformado por varios collider con diferentes características), este HollowObbCollider es el utilizado en cañón para hacer que la pelota ingrese en el mismo, para luego ser capturado por otro collider en el centro del cañón. De esta manera se permite cargar el cañón colocando la pelota en su interior para que este luego la dispare.
- *Meshes*: La mayoría de las meshes siguen al ítem en todas sus transformaciones, sin embargo algunas no como la base del cañón, que al igual que uno de sus tantos colliders, no rotan con el siempre. Además algunos objetos tienen más de una mesh y separadas, como es el caso del portal que posee una mesh (con una textura azul) la cual es el umbral de entrada y otra mesh (con una textura naranja) el cual es un umbral de salida y esta segunda no sigue a la primera ni en su textura ni en su color, es prácticamente independiente de la primera.
- *Particles*: A la hora de realizar los efectos de partículas (explicados más adelante), diferentes ítems poseían diferentes efectos, entonces, como muchos de estos efectos deben acoplarse al ítem en sus transformaciones o deben ocurrir en situaciones especiales y seguir al ítem, se decidió que estos Quads, sean parte del ítem en sí.

Las Meshes:

Todas las meshes utilizadas (incluso las paredes y la pelota) fueron realizadas en 3DMax y exportadas al TgcViewer con el importador provisto por la catedra.

Las Texturas:

Algunas de las texturas utilizadas fueron sacados delas que se encuentran en el TgcViewer, sin embargo la gran mayoría, se buscaron en internet, intentando que se mapee correctamente con el objeto que intentábamos lograr. Sin embargo como muchos objetos fueron realizados en el 3DMax, no se necesitaron tantas texturas.

A la hora de elegir las mismas, se decidió una temática espacial-futurista dado que muchos de los objetos (agujero negro, portal, repulsor) daban la impresión de un tiempo futuro.

Efectos de Sonido:

A la hora de ponerle sonido al juego, buscamos distintos sonidos en internet, aunque muchos nos sirvieron (como el disparo del cañón) muchos de ellos fueron creados por nosotros utilizando la herramienta provista por esta página: http://www.superflashbros.net/as3sfxr/

Esa herramienta te permite obtener diferentes sonidos editando ciertos efectos y frecuencias, logrando sonidos más acordes al ambiente que habíamos armado para el ejemplo.

Como música de fondo optamos por parte de la banda sonora de un juego conocido de PlayStation "Crash bandicoot 2", en donde la música con un estilo futurista que encaja con la temática elegida intentando agregar un ambiente más relajado o misterioso para los niveles que presentamos.

Efectos de Partículas:

Para lograr los efectos de partículas (el humo del cañón, el polvo de las paredes, las chispas del imán y el repulsor y el poder del agujero negro), conseguimos diferentes Atlas Textures (algunos por internet, otros hechos por nosotros con Gimp y Photoshop). Decidimos utilizar esta técnica ya que al poseer una jugabilidad 2D el efecto visual logrado era el deseado.

Para lograr el efecto, se necesitaba un Quad, al cual se le cargaba la textura y cambiando el mapeado de la misma (UV offset) a gran velocidad, se produce el efecto visual de la animación.

Uno de los problemas que se nos presento es que el TgcQuad solo permite setear un color (no puede ponerse textura) y el TgcPlaneWall solo tiene orientaciones paralelas a los ejes cartesianos (no nos permitía rotar la animación con nuestros objetos). Para resolver esto, reutilizamos el código de ambas clases del TgcViewer y definimos dos clases, TextturedQuad.cs y AnimatedQuad.cs, con estas dos clases logramos tanto el seteo de la textura como el mapeado de la misma durante la animación. Controlando el tamaño de la textura y la velocidad de la animación.

Además en aquellos efectos que dependen del rebote de la pelota, el tamaño del efecto tiene una relación directa con la velocidad de colisión de la misma, así logrando un efecto un poco más creíble.

Animaciones:

Las diferentes animaciones que se utilizaron, el encogimiento del resorte, el reposicionamiento del cañón, el cambio constante del agujero negro, decidimos aplicar diferentes tipos de transformaciones a los ítems, así rotándolos o escalándolos en diferentes momentos y de la manera adecuada, se logró el efecto deseado. Algunas animaciones (como las flechas del acelerador) se utilizaron la misma técnica que se usó en las partículas, logrando un movimiento continuo y constante sin modificar la mesh en sí.

Optimización:

En cuanto a la optimización del juego, uno de los problemas más grandes que tuvimos es la carga de los niveles, al levantarlos del disco (desde los archivos .xml) en el momento de inicio del juego, se perdía mucho tiempo en el inicio del juego. Para dar una solución a este problema, por un lado se decidió poner una pantalla de carga al inicio con una animación (para que el jugador vea que ya se inició el juego) y además, a la hora de levantar los niveles, estos no se levantan todos al inicio, sino que se levantan dinámicamente uno a uno, mostrándose el primer nivel una vez que ya se cargó. Para esto se utilizó un hilo secundario en la aplicación que se encarga de verificar si tanto el nivel siguiente como el anterior están cargados y en caso de no estarlos comienza su carga, pero sin interferir a la jugabilidad del juego.

Otro de los puntos en los que se intentó optimizar el ejemplo es en la cantidad de cálculos realizados durante la simulación física. Como la cantidad de meshes en pantalla no son demasiadas, el motor físico era una de las pocas cosas que enlentecían el juego. Para evitar el exceso de cálculos, se separó la simulación en diferentes etapas.