Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación

Informe Tarea 2

Creando un juego en 2D

Autor: Lukas Pavez
RUT: 19.401.577-1
Profesor: Nancy Hitschfeld K.
Auxiliares: Pablo Polanco Galleguillos
Pablo Pizarro R.
Mauricio Araneda H.
Ayudantes: María José Trujillo Berger
Iván Torres
Fecha de entrega: 08/07/2018

Índice de Contenidos

Lista de Figuras

1. Introducción

Bomberman es un juego en el cual un hombre-robot debe atravesar un laberinto al tiempo que evita a diversos enemigos. Las puertas que conducen a otras salas del laberinto se encuentran bajo rocas que Bomberman debe destruir con bombas. Hay objetos que pueden ayudar a mejorar las bombas de Bomberman, como la habilidad de fuego, que mejora el alcance de las explosiones de las bombas.

En esta tarea se realizará una implementación del juego Bomberman donde lo mínimo que se pide es: diseño de los objetos, que los personajes se muevan, choquen con los muros y pongan bombas que explotan luego 3 segundos, si el personaje principal toca una explosión o un enemigo el juego termina, debe haber una escena estática que cuente con muros indestructibles y una salida, muros destructibles que aparezcan de forma aleatoria en la escena, y también debe haber 2 power ups escogidos por uno. El resto de las cosas que se implementen es considerado bonus.

2. Diseño de la solución

Para la creación del juego se utilizó el diseño de MCV (modelo-vista-controlador), donde cada parte hace lo siguiente:

Controlador: inicializa todos los objetos, contiene el loop principal, analiza las colisiones entre los objetos, obtiene la información del usuario y le dice a la vista que tiene que dibujar.

Vista: elimina los objetos que ya realizaron sus interacciones (por ejemplo las bombas luego de explotar), y dibuja en la pantalla los objetos disponibles en sus posiciones respectivas.

Modelo: son los objetos que se utilizan durante el juego, los modelos creados son los siguientes:

- Fondo: corresponde a un rectángulo del tamaño de la pantalla de color verde, con un cuadrado rojo en la mitad del lado inferior, que corresponde a la salida por la que el jugador debe pasar para ganar.
- Bomb: corresponde a la bomba que ponen los personajes, contiene los métodos que hacen que 'explote' luego de 3 segundos, el diseño se puede ver en la Figura 1.
- Wall: corresponde a las murallas del juego, el diseño se puede ver en la Figura 2.
- BreakableWall: corresponde a las murallas del juego que se pueden destruir con las bombas, luego de crearla se pensó en juntar BreakableWall y Wall en un solo modelo, pero se decidió dar mas prioridad a agregar mas cosas al juego, el diseño se puede ver en la Figura 3.
- VelPowerUp: corresponde al power up que aumenta la velocidad de movimiento del jugador, el diseño se puede ver en la Figura 4.
- BombPowerUp: corresponde al power up que disminuye el tiempo en que el jugador puede poner una bomba, al principio puede poner una bomba cada 3 segundos, después del power up pone 1 bomba cada 1 segundo, el diseño se puede ver en la Figura 5.
- Bomberman: corresponde al personaje utilizado por el jugador y también a los enemigos, tiene métodos para moverse y poner bombas, el personaje utilizado por el jugador tiene la particularidad de levantar y bajar los brazos mientras se está moviendo, los diseños para los robots fueron inspirados en los robots del laboratorio de robótica de la universidad, el jugador utiliza a Bender (ver Figura 7) y los enemigos pueden ser naos o peppers (ver Figuras 8 y 9)

Para la construcción de la escena estática del juego primero se dibuja el Fondo, y luego se crean los suficientes objetos Wall que rodeen toda la escena, que tiene un tamaño de 800x600, con la excepción de 2 muros que van al medio en el lado inferior de la escena, estos 2 muros se reemplazan por un BreakableWall (el lado de un BreakableWall es de dos veces el lado de un Wall), esto es para crear la salida por la que el jugador debe pasar para ganar el juego, finalmente se crean bloques de 4 Wall cada uno y se ponen dentro de la escena, el diseño de la escena construida se puede ver en la Figura 6

3. Explicación detallada de la solución

El juego empieza cuando el controlador (main.py), llama a la función main(), esta función parte por crear una ventana de 800x600, crea un objeto Vista y setea la musica del juego, luego comienza a crear las listas con los objetos a utilizar, primero crea la lista con los muros para la escena estática, luego crea la lista con los muros destructibles en posiciones aleatorias, esto se hizo creando un muro en cada una de las posiciones disponibles con probabilidad del 20 %, esto significa que en cada posición, solo si math.random() es menor a 0.2 se crea el muro. Luego crea al personaje en la esquina inferior izquierda de la escena, y 4 enemigos en posiciones aleatorias disponibles, los diseños de los enemigos se van alternando por cada enemigo que se crea, y por último crea 2 power ups en cualquier posición de la escena donde no haya un muro indestructible, por lo que las posiciones disponibles pueden ser donde aparezca un enemigo o un muro destructible.

Después de crear todas las listas de objetos continúa con el loop principal del juego, donde primero se revisa si se ganó el juego, esto se hace revisando el atributo win de bomberman, que es True si el jugador llega a la salida, también revisa si se pierde el juego viendo el atributo active del jugador, si es False entonces se pierde el juego, que ocurre cuando el jugador recibe una explosión o choca con un enemigo, después revisa las las teclas que presiona el usuario y deja en True el atributo del bomberman de la dirección presionada (ej: se presiona la flecha derecha, entonces bomberman.move_right=True) y el resto de las direcciones se dejan en False, para no moverse en diagonal se selecciona una de las dos direcciones, si se presiona la tecla q el juego termina, y si se presiona la tecla a se agrega una nueva bomba a la lista de bombas, luego se hace que todos los enemigos intenten poner una bomba, cada uno tiene probabilidad de 0.5 % de ponerla, si lo hacen se agrega la bomba a la lista de todas las bombas, luego se mueven los personajes, para esto el bomberman se mueve según la dirección que tenga como True, donde se le suma al vector de posición el vector de velocidad del eje en el que se va a mover ponderado por un dt que es positivo si se mueve hacia la derecha o hacia arriba y negativo si se mueve en las otras direcciones. Para que los enemigos se muevan se hace un random, 20 % se queda quieto y 80 % se mueve, si es que se mueve se hace un random para seleccionar una de las 4 direcciones con la misma probabilidad cada una, luego se llama al método mover del enemigo que es el mismo método del bomberman.

El programa continua revisando las colisiones, primero se revisan las colisiones de todos los jugadores con los muros, si un punto del rectángulo que contiene al personaje está dentro de un punto del rectángulo que rodea al objeto con el que choca, el personaje mantiene la posición en la que estaba, esto también ocurre cuando un personaje choca con una bomba antes de que explote, con la diferencia de que se tiene un tiempo para moverse por encima de la bomba para que no choque con ella inmediatamente al ponerla, ya que la bomba se pone en la misma posición en la que se encuentra el personaje. Continúa revisando las colisiones entre bomberman y los power ups, cuando chocan se llama el método trigger del power up con el que choca, este método hace una modificación en los atributos del bomberman y luego cambia su estado active que estaba en True a False. Después se revisan las interacciones de las bombas que están explotando, si uno de los jugadores choca con el rango de colisión de la bomba, entonces el estado active del jugador pasa a ser False, lo mismo ocurre con los muros destructibles, si uno choca con el rango de explosión, su estado active pasa a ser False. Finalmente se revisa si un enemigo choca con el bomberman, si esto ocurre entonces el estado active del enemigo y del bomberman pasa a ser False.

Finalmente el controlador llama al método explode de todas las bombas, donde cambia el

atributo exploding a True, donde si se acaba el timeout de explosión (3 segundos), y si ya estaba explotando espera otro timeout (1 segundo) y cambia el estado active de la bomba a False, y después llama al método dibujar del objeto vista creado al principio donde le entrega todas las listas con los objetos a dibujar, estos son los jugadores, el fondo, las murallas, los power ups y las bombas. Lo primero que hace el método dibujar es "limpiar"las listas que les entregan, esto es revisar las listas de las bombas, muros destructibles, power ups y jugadores, y elimina de las listas los objetos cuyo atributo active sea False, y luego de actualizar estas listas llama al método dibujar de todos los elementos de las listas, partiendo por el fondo, luego los muros indestructibles, los power ups, los muros destructibles, las bombas y los jugadores, el orden es importante ya que el último que se dibuja queda encima del resto, por eso se dibujan los power ups antes que los muros destructibles, ya que puede aparecer uno debajo de un muro.

4. Discusión sobre las dificultades encontradas

La primera dificultad encontrada al hacer la tarea fue no saber como empezar, esto se soluciono tomando la tarea Icy Tower que uno de los auxiliares subió a u-cursos y leerla hasta que se empezó a entender cómo funcionaba, después lo primero que hice en mi tarea fue ver como dibujar un modelo y como moverlo con las flechas, cuando se logró esto ya se pudo seguir con los otros modelos y la construcción de la escena, hasta que se llegó a lo que pienso que tenía la mayor dificultad: las colisiones, si bien esto me tomó bastante tiempo al final se pudo resolver, el resto del juego no presentó gran dificultad pero si tomo tiempo realizarlo, ya que antes de agregar cosas había que entender como funcionaban.

5. Aprendizajes obtenidos

Aparte de las dificultades encontradas, la tarea resulto entretenida, y me sirvió para aprender como leer algo que ya esta hecho (el Icy Tower) y como entenderlo para aplicarlo en mi propia tarea, tambien me sirvió aprender MVC ya que en la tarea 3 de otro ramo voy a tener que hacer la interfaz gráfica de un juego (la tarea 2 fue hacer los modelos y el controlador), y otra cosa que se aprendio es a usar python 3 (nunca antes lo había ocupado), ya que la tarea sobre la que me basé para hacer esta (Icy Tower) estaba en ese lenguaje de programación.

6. Anexos

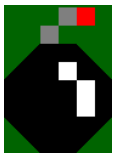

Figura 1: Diseño utilizado para las bombas

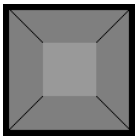

Figura 2: Diseño utilizado para las murallas indestructibles

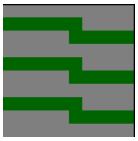

Figura 3: Diseño utilizado para las murallas destructibles

Figura 4: Diseño utilizado para el power up de velocidad

Figura 5: Diseño utilizado para el power up de bombas

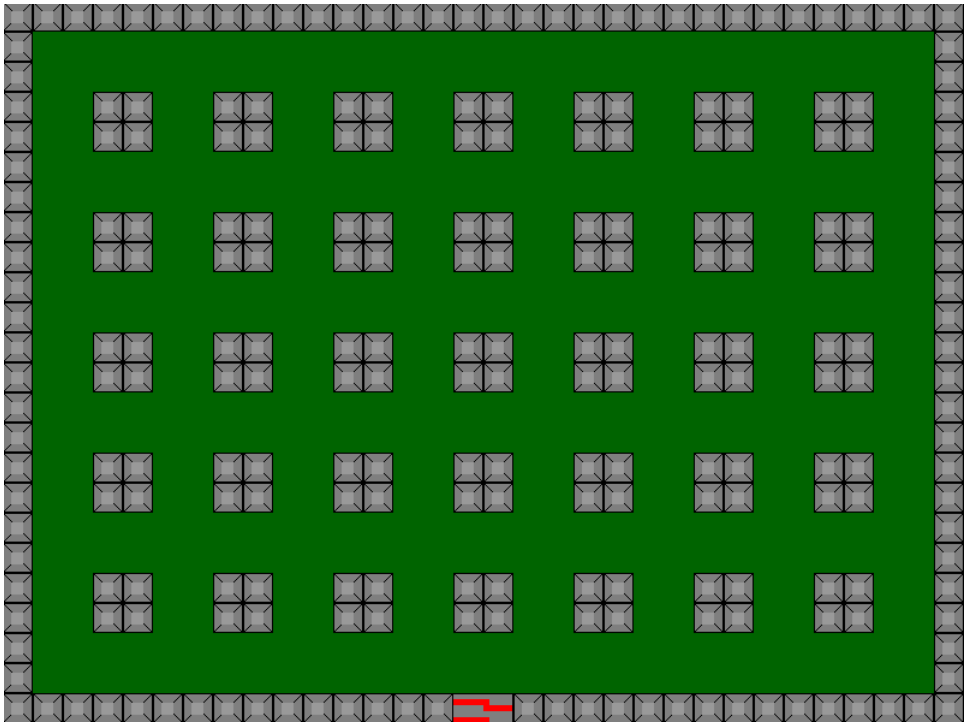

Figura 6: Diseño utilizado para escena estática del juego

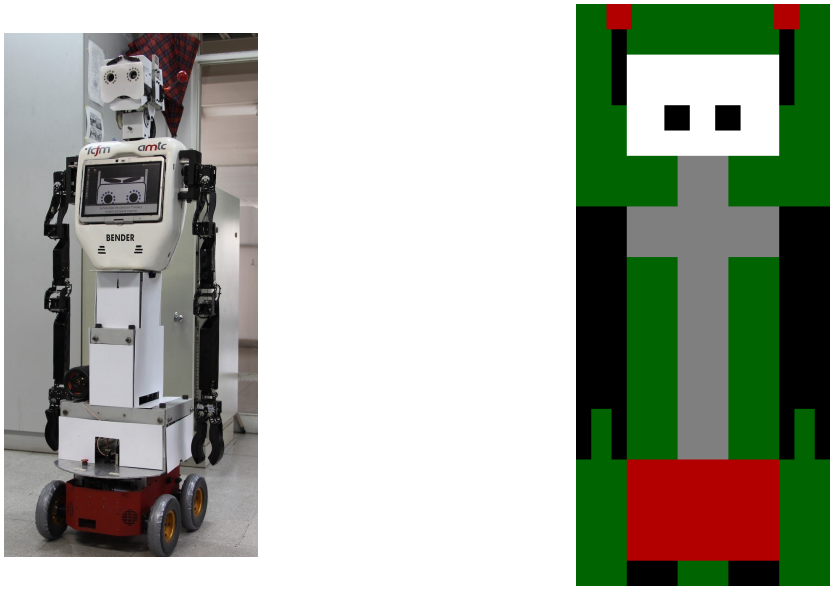

Figura 7: Robot Bender y diseño utilizado para el juego

Figura 8: Robot nao y diseño utilizado para el juego

Figura 9: Robot pepper y diseño utilizado para el juego