Departamento de Ciencias de la Computación
Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
CC3501-1 Modelación y Computación Gráfica

SPACE WAR

Tarea 1: OpenGL2D

Alumno: Alfredo Escobar
Profesor: Daniel Calderón
Auxiliares: Nelson Marambio Q.
Alonso Utreras
Ayudantes: Tomás Calderón R.
Nadia Decar
Beatriz Graboloza M.
Heinich Porro Sufan
Fecha: 9 de mayo de 2020

1. Solución Propuesta

1.1. basic_shapes.py

Se ha creado la función "createColorRombo(r, g, b, half=0)", que entrega un rombo de color para ser utilizado el la construcción de las naves. La función "createColorPoligono(r, g, b, n)" entrega un polígono regular de n lados, se utiliza para crear los "círculos" de los planetas y estrellas. Finalmente, "createColorTriangle(r, g, b, punta)" entrega un triángulo de color, en función de la posición del vértice entregado.

1.2. game_shapes.py

Primero se han creado funciones que fabrican los nodos de GPUShapes de la nave del jugador, las naves enemigas, y los planetas. Luego, se han escrito funciones de instancing para las varias posibles existencias de naves enemigas, planetas, estrellas, láseres y cuadros de puntos de vida del jugador.

1.3. space-war.py

Se ha escrito la función "moverPlayer" que, como su nombre lo indica, se encarga de mover la nave del jugador según las teclas presionadas.

La función "moverEnemigos" tiene distintos propósitos, dependiendo de la posición de cada una de las naves enemigas:

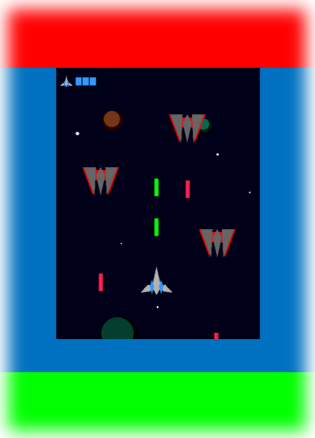

Figura 1: Diagrama para "moverEnemigos"

Si uno de los enemigos se encuentra en la zona roja o en el interior de la ventana, la función se encarga de hacer avanzar a la nave hacia abajo y hacia los lados. Si el enemigo se encuentra en la zona azul, la función se encarga de moverlo hacia un punto aleatorio de la zona roja y asignarle una dirección de viaje aleatoria. La función ignora las naves en la zona verde (esta zona se utiliza para "guardar" las naves en desuso).

La función anterior también se encarga de manejar las colisiones entre la nave del jugador y las naves enemigas (esto se profundizará a continuación).

La función "moverLasers" se encarga de el avance y la aparición de todos los láseres. También se encarga de las colisiones entre los láseres del jugador y los enemigos, y de las colisiones entre los láseres de los enemigos y el jugador.

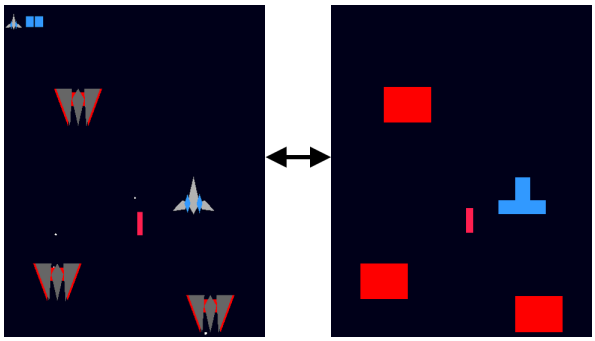

Figura 2: Diagrama de hitboxes

Las colisiones se evalúan midiendo las diferencias horizontales y verticales entre los centros de los dos objetos a analizar. Las distancias horizontales y verticales al centro de cada objeto, en las cuáles se pueden detectar colisiones, forman un área llamada hitbox. Estas hitboxes están representadas en la figura superior. La detección de colisiones de la función "moverEnemigos" funciona del mismo modo. Cada vez que una nave enemiga recibe un impacto, es enviada a la zona azul señalada en la Figura 1, para que se le sea asignada una nueva posición aleatoria en la zona roja.

Es importante señalar que, internamente, existe una cantidad limitada de láseres (5 para el jugador, y 3 para cada uno de los enemigos), por lo que cíclicamente se elige cuál será el siguiente láser en salir disparado de cada nave, para dar la impresión de "láseres infinitos".

La función "gameProgress" se encarga de mover los cuadros de la barra de vida fuera de la pantalla cada vez que el jugador recibe un impacto. Además, mueve las texturas de finalización de la partida (los textos "Congratulations" y "Game Over") hacia el centro de la pantalla cuando corresponda.

Para el manejo de los elementos de fondo (planetas y estrellas), se crean dos nodos de cada tipo. Uno se dibuja dentro del área de la ventana, mientras que el otro se dibuja arriba del primero. Ambos nodos avanzan juntos hacia abajo, y cuando el primer nodo ha salido de la pantalla, es llevado arriba del otro nodo. Este proceso se repite indefinidamente. Tanto estrellas como planetas son generados con valores aleatorios para tamaño y posición (y color en el caso de los planetas).

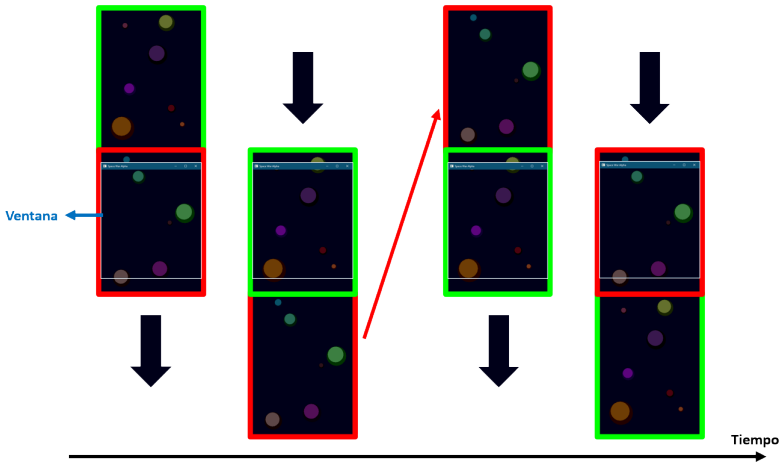

Figura 3: Diagrama de movimiento de elementos del fondo

Finalmente, existen matrices tridimensionales, llamadas “rojos_vars” y “lasers_vars”, que manejan las posiciones y propiedades de cada una de las naves enemigas o de cada uno de los lásers, respectivamente. Por ejemplo, en “lasers_vars” existen, por cada nave, valores que indican cuál será el siguiente láser en ser disparado (recordar que los láseres como nodos de GPUShapes son limitados en cantidad), o el momento en que dicho láser va a ser disparado (en el caso de los láseres enemigos).

2. Instrucciones de Ejecución

2.1. Argumentos

El programa se ejecuta con la siguiente llamada:

```
python space-war.py N
```

En donde N representa la cantidad total de naves enemigas que aparecerán en el transcurso de la partida.

2.2. Teclas de control

El movimiento de la nave del jugador se realiza al mantener presionadas las teclas “A” (izquierda), “S” (atrás), “D” (derecha) y “W” (adelante). Al presionar más de una de estas teclas, se permite el movimiento en diagonal.

Para disparar láseres desde la nave del jugador, basta con presionar la tecla “ESPACIO”. Esta tecla debe ser presionada para cada disparo, pues el juego no admite disparo automático (mantener presionada la tecla no disparará varios láseres seguidos).

Finalmente, se puede cerrar el juego mediante la tecla “ESCAPE”

3. Resultados

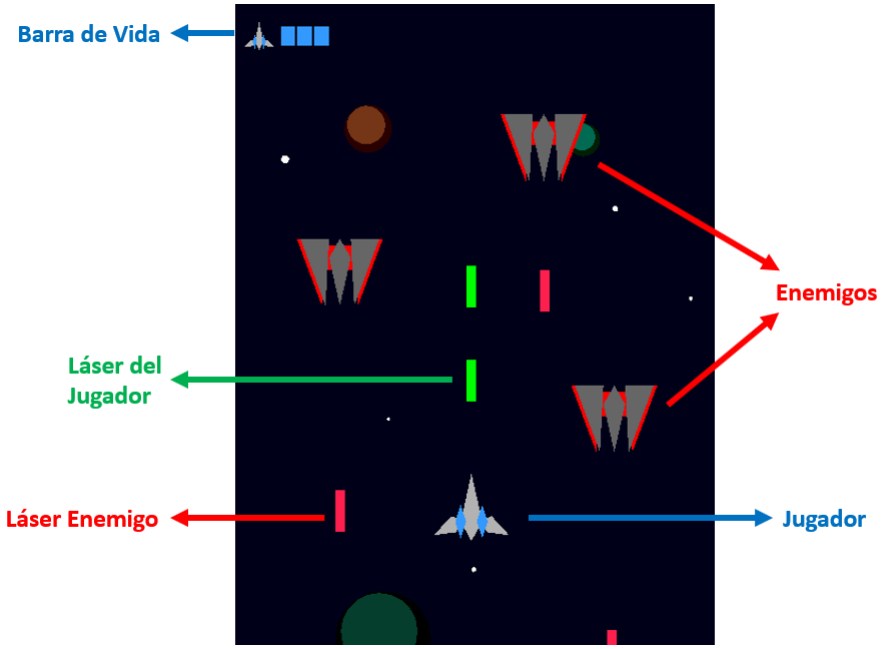

Figura 4: Elementos de la partida

El objetivo del juego es destruir a todas las naves enemigas con los lásers del jugador y, al mismo tiempo, evadir los disparos de estas.

Cada nave enemiga aparecerá desde la parte superior de la ventana, avanzando hacia la parte inferior. Dispararán lásers rojos periódicamente.

Al inicio del juego se mostrará una sola nave enemiga en pantalla. A los 5 segundos aparecerán 2 naves simultáneamente, y a los 10 segundos serán 3 los enemigos que aparezcan al mismo tiempo en pantalla.

En la esquina superior izquierda de la ventana está la barra de vida de la nave del jugador. Sus cuadros indican su vida restante. Cada vez que el jugador reciba el impacto de una nave o láser enemigo, desaparecerá uno de los cuadros. Si la nave recibe 3 impactos, se acabará la partida.