Curso: Modelamiento y computación grafica para ingenieros
Profesor: Daniel Calderón
Alumno: Karen Lavados jilbert
Fecha: 30 de abril 2020

Reporte de Tarea1

Este reporte describe la arquitectura realizada para solucionar la tarea 1 del curso CC3501-Modelamiento y computación grafica para ingenieros. En esta ocasión la opción seleccionada fue la opción b, la cual consiste en la creación de un juego interactivo llamado “Space Wars”, el cual se desarrolló con lenguaje Python.

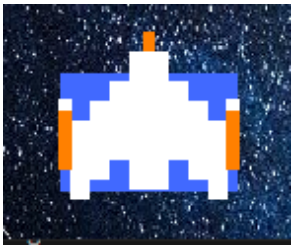
Solución propuesta

Para poder almacenar múltiples datos de las naves enemigas como de la principal, se decidió crear la clase Nave. Esta clase tiene diversos atributos como la vida, la posición de la nave como la de sus rayos, el área interna de la nave entre otras cosas. Además de almacenar estos datos tiene distintas funciones en ella, las principales son , la función ataque () la cual se encarga de disparar el rayo, la función colisión() la cual se encargará de detectar si la nave ha recibido algún daño , la función ciclo() que se encarga de darle movimientos a las naves y la función Jesús(), que se encargará de resucitar a la nave ,reiniciando sus datos para que vuelva al juego. Todas las funciones anteriores son usadas exclusivamente por las naves enemigas , ya que ,la nave principal pose sus propias funciones dentro del while de la función main(), pero son análogas a las mencionadas anteriormente , solo presentan algunas diferencias en cuanto a código.

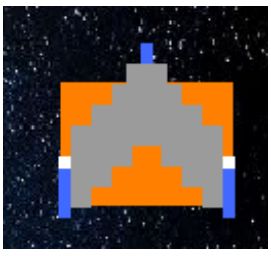
La nave principal es controlada con las teclas WASD y disparara los rayos presionado el botón Space, cabe mencionar que la nave principal solo tiene la capacidad de lanzar tres rayos consecutivos , por lo que, si se aprieta más de tres veces la tecla y los tres rayos aún siguen en el juego , no saldrá ninguno nuevo. En cuanto a los rayos de las naves enemigas , estas dispararan un rayo cada cierto periodo de tiempo automáticamente.

Los modelos de las naves enemigas y la de la principal se realizaron utilizando múltiples transformaciones y juntándolas hasta dar con la forma deseada. El rayo se realizó de forma similar, pero este resulta ser un cuadrado escalado para que se asemeje a un rectángulo.

Anteriormente se mencionó que la clase nave tiene un atributo llamada área interna, esto consiste en un cuadrado que se ajusta a la nave y que se va moviendo continuamente junto a ella . Esta área servirá para detectar si un rayo colisiono con la nave , porque solo bastará ver si el rayo entro a esta área. A continuación, se mostrará el modelo de las naves y una simulación de sus áreas internas.



Nave Principal



Nave enemiga

Curso: Modelamiento y computación grafica para ingenieros
Profesor: Daniel Calderón
Alumno: Karen Lavados jilbert
Fecha: 30 de abril 2020

El fondo del juego se encuentra en continuo movimiento lo que da una sensación de que la nave va avanzando en tiempo. Para general este movimiento se adapto la imagen a la pantalla y se creó un ciclo para que continuamente mostrara la secuencia de una imagen normal y luego la misma imagen, pero reflexionando en torno al eje Y. Además de esta imagen se usaron otras dos, una que dice GameOver, la cual aparecerá en el momento en que la nave principal reciba su tercer daño, mostrando que el jugador perdió .En cambio, la segunda imagen aparecerá cuando el jugador logra destruir todas las naves enemigas , la cual mostrara la imagen con las palabras " you win".

Al iniciar del juego la nave principal se encontrará en la parte inferior y las naves enemigas aparecerán paulatinamente del extremo superior izquierdo de la pantalla cuando estas sean requeridas . En lo posible siempre habrá 3 naves enemigas en el sector superior de la pantalla , si se llega a destruir una y aún quedan naves por salir , reaparecerá una nave cuando esta lo pueda realizar sin solaparse con alguna otra nave enemiga.

Instrucciones de ejecución

Como se mencionó anteriormente, la nave principal se podrá mover con las teclas WASD del teclado y disparara usando la tecla Space, las naves enemigas se moverán y actuaran automáticamente. La cantidad inicial de naves dependerá de cuantas el usuario haya decidido ingresar , esto se realizará agregando un numero al momento de ejecutar el programa en anaconda como se puede apreciar en la imagen , el cual muestra un ejemplo con 10 naves enemigas.

```
(base) C:\Users\karen>conda activate python-cg

(python-cg) C:\Users\karen>cd C:\Users\karen\OneDrive\Escritorio\GitLab\lavados-jilbert_karen-constanza

(python-cg) C:\Users\karen\OneDrive\Escritorio\GitLab\lavados-jilbert_karen-constanza>python Tarea1b.py 10
```

Resultados

A continuación, se procederá a mostrar sreenshots del juego desarrollado en esta tarea, las cuales son cuando el jugador logra derrotar a todos lo enemigos ganando el juego , cuando el jugador ha

recibido 3 daños por parte de las naves enemigas, por lo tanto su nave a sido destruida y perdió el juego y por último se muestra una imagen donde el juego se encuentra en pleno funcionamiento

Imagen cuando el jugador pierda

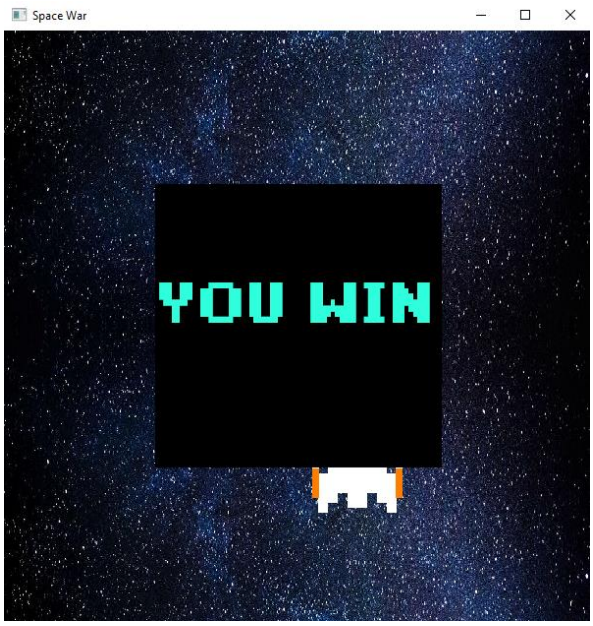


Imagen cuando el jugador gano

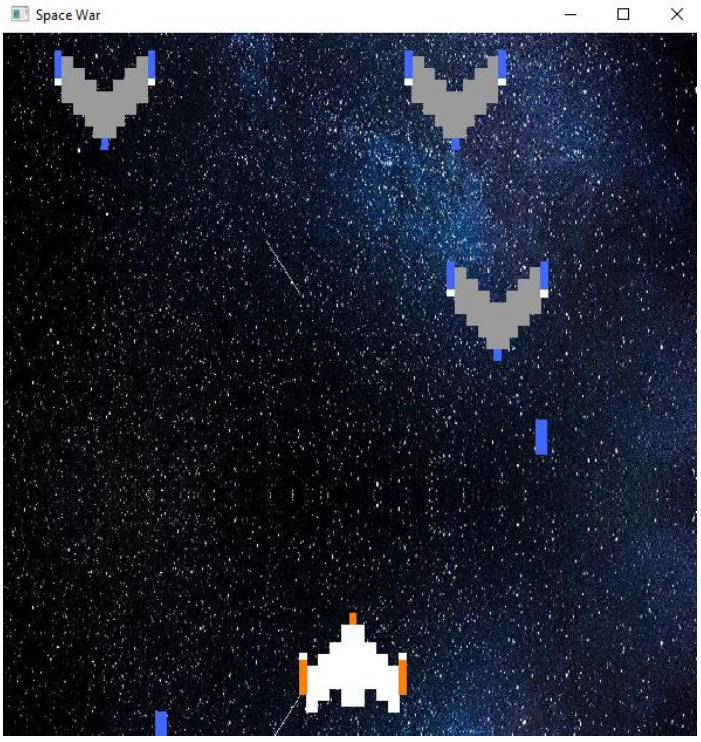


Imagen del juego en pleno funcionamiento