Segundo proyecto programado

I Semestre 2018

Kevin Zeledón Salazar
Esteban Medina Navarro

20/5/18

Índice

1.Introducción

En el segundo proyecto programado se va a hacer una versión del clásico juego Pong, lanzado en 1972 por Atari. Pong se basa en el deporte llamado tenis de mesa (ping pong). El juego se basa en dos dimensiones donde un jugador controla una paleta que devuelve una pelota y el objetivo es vencer al oponente (ya sea otro jugador o la máquina) haciendo que este no pueda devolver la pelota. Para crear el juego se mezclaron dos recursos: para el menú se utilizó el ya antes usado tkinter y para el juego en sí se utilizó pygame.
Al crear un juego en este se le incita a los programadores a investigar y experimentar como es el desarrollo de estos, ya que el mercado de la programación es abarcado en gran medida por los videojuegos.

2.Descripción del problema

Este segundo proyecto programado es realizado en parejas por lo que el uso de GitHub es requisito para este proyecto.
Además antes de iniciar el trabajo de programación se le da prioridad a definir el modelo de objetos para el proyecto, tomando en cuenta las funcionalidades que debe tener el juego para así llevar un mayor orden a la hora de trabajar en el grupo y distribuir las tareas.
Como la presentación de interfaz es libre se decidió utilizar tkinter para el menú y para el juego en sí se utilizó pygame. Una vez en el menú y en el juego se encuentra las funciones y características requeridas como:

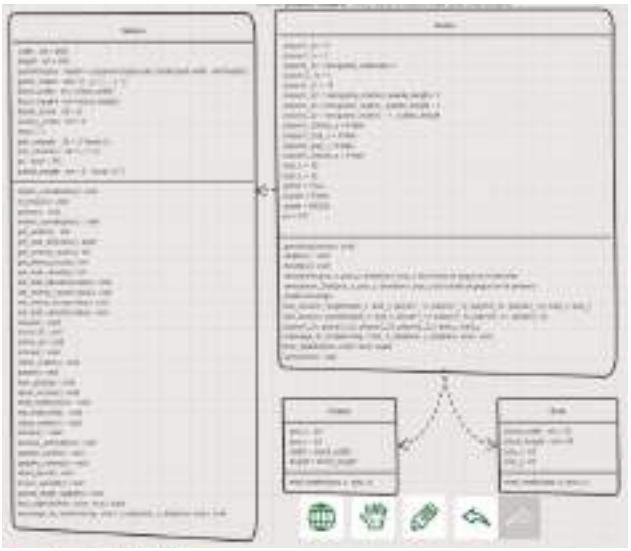
- Modos de juego:
    - Contrincante: Un requisito es que el jugador pueda elegir si su contrincante va a ser manejado por otro jugador o por la máquina. Esto se encuentra en el menú principal
    - Cantidad paletas: El jugador debe elegir si juega la modalidad clásica de una paleta o juega con dos paletas que consiste en que cada jugador va a poseer dos paletas que se mueven con el mismo control de movimiento
- Dificultad dentro del juego: En ambas modalidades la dificultad aumenta de manera diferente:
    - Player vs player: La dificultad aumenta cuando se lleva una cantidad de tiempo sin hacer un punto por parte de ninguno de los jugadores, hasta llegar a la dificultad 3. Al anotar un punto vuelve a la dificultad 1.
    - Player vs PC: Hay 3 rondas de 10 puntos, al ganar una ronda se pasa de nivel hasta ganar la dificultad 3, lo que significa que ganó el juego.

- Funcionalidad de las paletas: Las paletas deben funcionar de manera que si la pelota golpea en la parte superior el rebote es hacia abajo, si es en el medio la pelota se devuelve horizontalmente y si el golpe es arriba la bola irá hacia abajo.
- Matriz: Debe existir una matriz que represente todo lo que esta pasando en el juego, elegimos hacerla una matriz booleana por la simplicidad del juego, donde True representa un cuadro blanco de 20x24 y False un cuadro negro de las mismas dimensiones. Además, todas las colisiones se hicieron a mano siguiendo éste mismo sistema.

Además de características y funciones extra como:

- Help: instrucciones para la comprensión del usuario
- Adición de jugador 2 en medio juego vs pc.
- Música diferente por nivel en un pseudo modo campaña.
- Pantalla de pausa.
- Ciclicidad (al terminar un juego se puede empezar otro, igual o diferente)
- En doubles el movimiento de las paletas es opuesto.

3.Diagrama de clases



Link de la pizarra con el diagrama: https://sketchboard.me/vA1i65sRLLte#/

4.Análisis de resultados

El programa se encuentra dividido en dos archivos de python uno con la interfaz como se ve en la imagen 1, la cual está hecha en tkinter. Esta tiene la funcionalidad de abrir el juego en la modalidad player vs player y player vs PC, el cual puede cambiar si se une un nuevo jugador a la partida. Estas a la vez pueden ser ejecutadas en modo singles o doubles.

Además de tener una ventana extra que muestra una breve descripción del juego con sus respectivas instrucciones de juego.

Imagen 1.Menú

En la imagen 2 se muestra ya lo que es el archivo del juego ejecutado en Pygame, en el modo singles en el cual cada jugador posee una paleta.

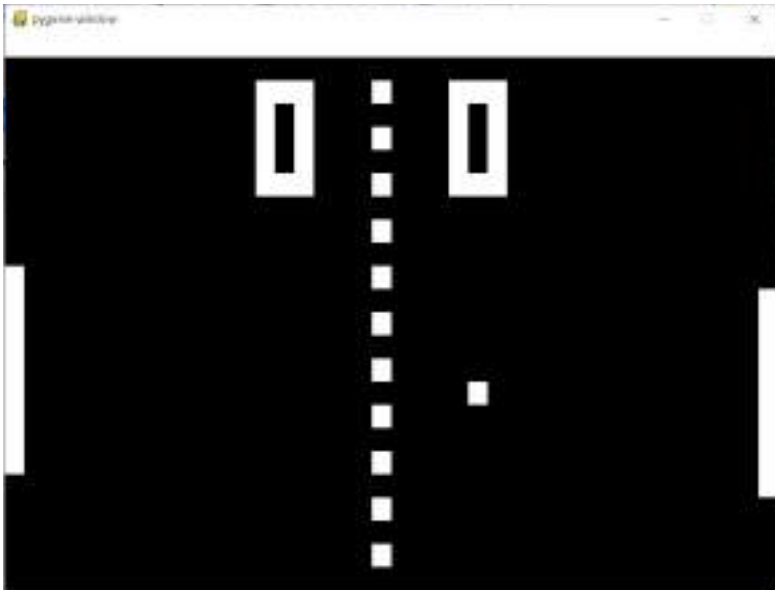


Imagen 2.Singles

El modo doubles como se muestra en la imagen 3, únicamente difiere de singles en que cada jugador posee 2 paletas controladas con los mismos controles que en single, de manera que una se mueve opuesto a la segunda.

.

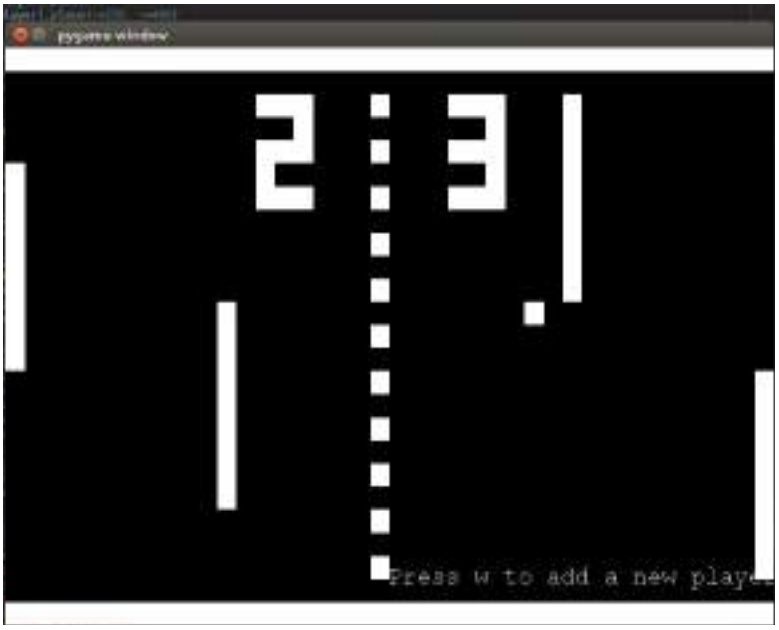


Imagen 3.Doubles

Además dentro del juego va aumentando la dificultad dependiendo del modo actual ya que en pvp aumenta conforme pasa el tiempo sin que ningún jugador anote un punto y al anotar vuelve a la normalidad y en pvpc son 3 rondas de diferente dificultad contra la pc de 11 puntos cada una.
El trabajo con clases se realizó empezando por el diagrama de clases (3.Diagrama de clases) para de esta manera realizar un mejor reparto de tareas y diseñar el juego con mayor orden. Las clases fueron:

- Tablero
- Game
- Paleta
- Bola

Además se usaron otros recursos importantes como mutagen, las librerías sys, os, time y random a explicar en el apartado 5.

5.Dificultades encontradas

A continuación se enlistan las dificultades encontradas con orden de aparicion:

1. Hitblock de la bola:
    a. La bola siempre ha sido un rectángulo de 24 x 20 pixeles en la pantalla. Al principio lo que se hacía para lograr el rebote era que cuando la posición en x y en y de la bola en la matriz fueran iguales a los de la paleta, se diera el rebote, sin embargo esto traía un problema, al venir desde arriba hacia abajo y pegar con la parte más alta de la paleta, tocaba la esquina inferior derecha (o izquierda) de la bola, pero no rebotaba. Para solucionar éste problema se

añadió a la condición de rebote no solo que la posición en x y y de la bola fueran iguales a la paleta, sino que según la dirección que llevara la bola se analizaran distintos casos de rebote para la misma.

2. Inteligencia artificial de la bola:
    a. Como puede notarse al jugar, se logró que la computadora tuviera un movimiento elegante, fluido, natural y posible de vencerse, sin embargo esto se logró después de muchas mejoras a la misma. La inteligencia artificial empezó siendo un algoritmo que seguía la posición en y de la bola, lo cual la hacía parecer muy mecanizada e incluso imposible de vencer, por lo que se empezó a reflexionar sobre el movimiento de la computadora en otros juegos de pong. Se notó que:
        i. La máquina empieza a moverse una vez que la bola vaya en su dirección, no antes.
        ii. La máquina parece saber donde va a caer la bola, pero falla por un poco algunas veces.

    De esta manera empezó la construcción de una versión mejorada de la inteligencia artificial, se creó un algoritmo que realiza una simulación instantánea del movimiento de la bola, tomando en cuenta su dirección y de donde fue lanzada, de esta forma ya la máquina sabe exactamente donde va a caer la bola. Una vez la máquina sabe esto se procede, mediante la librería random a generar un número random que oscila entre el negativo de la mitad del largo de la paleta y su positivo, de manera que el punto que calcula de esta manera nunca esta fuera de la pantalla y si lo esta, vuelve a calcularlo, de esta manera la inteligencia no es 100% invencible y tiene un cierto rango de error. Ahora la inteligencia procede a decidir, mediante random.choose(), con que lugar de la paleta va a golpear, tomando en cuenta que no puede salir ni un pixel fuera de la pantalla. Procede entonces a moverse tal como lo hace el humano, pixel a pixel por cada iteración de gameloop, con su objetivo siendo alcanzar que el lugar de la paleta que eligió alcance la posición en y donde calculó, con error a propósito, que va a caer la bola.
3. Rebote correcto:
    a. Inicialmente el rebote era errático, se solucionó mediante condiciones que evalúan donde se dió el golpe de la bola, dividiendo la paleta en tercios, donde el primer tercio es hacia arriba, el segundo tercio es directo y el ultimo tercio hacia abajo.
4. Música:
    a. Cualquier archivo de audio tiene entre su metadata una información que se llama sample_rate, que es como una medida de la velocidad a la que se debe reproducir la canción para que suena bien, éste dato lo ocupa Pygame para cada canción que va a reproducir, por las malas descubrimos que no lo hace correctamente. Se solucionó mediante una librería llamada Mutagen que toma un archivo de audio y retorna toda su metadata para que el usuario la pueda usar a placer.
5. Unión de interfaz con juego (bugs):
    a. Después de la primera unión de partes, para pasar de la ventana de tkinter a la de pygame se destruía la pantalla de tkinter y se crea una nueva instancia

de Game(), el problema con esto era que después de terminar el juego solo se cerraba pygame y la instancia no se destruía, pero sí se creaba otra y otra y otra, haciendo que a partir de la tercera instancia creada de Game(), se cayera el juego. Se solucionó mediante la creación de archivos aparte, uno para tkinter y otro para pygame, cada vez que se ocupa el archivo de pygame se utiliza el modulo os.

6.Bitácora de actividades

Kevin:

- 4/5/18 9:00 pm-2:00 am

Investigación de diagrama de clases
Diseño de diagrama de clases

- 5/518

Creación del archivo principal 9:30 am
Creación del tablero y campo de juego 10:31 am-12:15 pm
Creación de clases: single, bola y paleta 6:46 pm-11:20pm

- 6/5/18

Trabajo en la AI, adición de pausa 11:30 am-8:39 pm

- 7/5/18

Cambio de niveles tanto en singles como en doubles y resolución de problemas 6:00 pm-8:30 pm

- 8-9/5/18

Trabajo con la música 3:30 pm- 9:30pm
Condición de victoria, timer, arreglo de bugs 10:00 pm-12:00 pm

- 11/5/18

Arreglo de errores, arreglo en doubles 8:00 pm-10:30pm

- 12/5/18

Afinado de la AI 6pm-10pm

- 13/5/18

mejora de rebote,reset level 4:30 pm- 5:00 pm

- 16/5/18

Arreglo en bug de puntaje, mejoras en POO, solución en problemas de instancias 7:30 am-10:15 am

- 17/5/18

Adición de pantalla win y lose, mejora en la fluidez de la pelota 12:40 pm-9:20 pm

- 20/5/18

Parámetros añadidos en la clase Game, arreglo de paletas invisibles, niveles reiniciados al anotar en pvp y otros errores menores 12:19 pm- 12:00 am

Esteban:

- 4/5/18 10:00 pm-2:00 am

Investigación de diagrama de clases
Diseño de diagrama de clases

- 5/5/18 3:30 pm - 6:30 pm /9:30 pm - 12:00 am

Investigación de clases
Creación de clases

- 6/5/18 6:00 pm -10:30 pm

Diseño de interfaz
Creación de interfaz

- 10/5/18 2:00 pm-7:00 pm

Creación de interfaz
adición de sonido y botones
creación del toplevel help

- 11/5/18 2:00 pm-8:00pm

Arreglo de errores
Finalización del toplevel help

- 14/5/18 2:00 pm-5:00 pm

Trabajo con el sonido durante la pausa

- 16/5/18 2:00pm-4:00pm

Arreglos en la interfaz
Fusión de interfaz y juego
Audio en el lobby/menú principal

- 18/5/18 12:00 md-3:00 pm

Arreglo de bugs y problemas varios

- 20/5/48 12:00 md-12:00am

Cambio de root en función
Continúa unión de interfaz con juego
arreglo de bugs varios
Arreglos en las modalidades pvp, pvpc, singles y doubles, en sus ejecuciones
Asignación de modo singles como default
documentación interna
Paso de root() a otro archivo de python
arreglos en interfaz
arreglos en la ejecución de pygame

7.Estadística de tiempo

Función	Kevin	Esteban	TOTAL
Análisis de requerimientos	1.75 Horas	1.75 Horas	3 Horas
Diseño de la aplicación	4 Horas	4 Horas	8 Horas

Investigación de funciones	3 Horas	1.75 Horas	4.75 Horas
Programación	35 Horas	32.5 Horas	67.5 Horas
Documentación interna	1 hora	2 Horas	3 Horas
Pruebas	4 horas	1.5 Horas	5.5 Horas
Elaboración documento	3 horas	3 Horas	6 Horas
TOTAL	**51,75**	**46.5 Horas**	**98,25 Horas**

8.Conclusiones

Esteban:

- La creación de juegos genera gran cantidad de bugs (errores) o funciones no deseadas.
- Gran campo de la programación de hoy en día se enfoca a los videojuegos y crear uno nos muestra a los programadores una pequeña parte de lo que es crear un juego.
- Se aprecia más el trabajo de los programadores detrás de los juegos y también otros programas

Kevin:

- La inteligencia artificial no es un término tan increíble como se cree, es más bien muy simple.
- El programar videojuegos es una buena y divertida manera de probarse uno mismo.
- La música tiene metadatos que realmente son importantes.
-