Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

Proyecto 1
Simulador de Pool - Fase 1: Detección de Colisiones y simulación física base

El grupo docente de Algoritmos y Programación organiza al final de cada semestre una reunión que siempre concluye con un pequeño torneo de Pool.

El gran problema que se presenta con dicho torneo, es que para ser realmente sinceros, ningún integrante del grupo es un buen jugador.

Por lo tanto para este semestre se propuso conseguir alguna forma de entrenar sin necesariamente requerir una mesa de Pool real. En este punto es donde usted como programador del grupo de estudiantes de Algoritmos y Programación juega un papel importante.

Su trabajo tendrá como resultado final realizar una simulación con física básica de un juego de pool, sin embargo dicho trabajo será realizado en dos fases con el fin de facilitar su desarrollo.

Para esta primera fase se realizarán las funciones bases necesarias para luego propiamente poder realizar la simulación completa del juego. El conjunto de funciones requeridas serán detalladas más adelante en el documento, pero primero especificaremos las estructuras necesarias para poder representar la mesa y las bolas de pool.

1 Representación de la mesa

Como todos conocemos una mesa de Pool tiene particularidades que la diferencian de una mesa común, las cuales corresponden con una superficie de fieltro donde se colocan las bolas y un conjunto de agujeros o troneras donde deben ser colocadas las bolas con el fin de ganar el juego.

Para representar la mesa se le suministrarán dos puntos coordenados (x, y) que representan la esquina superior izquierda de la mesa y la esquina inferior derecha de la mesa. Con esta información usted debe ser capaz de representar la superficie de juego tomando en cuenta las siguientes reglas:

- El primer par de coordenadas corresponde con la esquina superior izquierda, lo que implica que el mismo debe encontrarse por encima y a la izquierda del otro par de coordenadas.
- El segundo par de coordenadas corresponde con la esquina inferior derecha, lo que implica que el mismo debe encontrarse por debajo del primero y a su derecha.

- Ambos pares de coordenadas están ubicados en un sistema de coordenadas tradicional centrado en el (0, 0).
- Una mesa de Pool cuenta con seis troneras, donde cuatro de las mismas se encuentran en las esquinas de la mesa y dos a los costados.
- Las troneras son representadas como un circulo centrado en las esquinas para el caso de las esquinas y en el punto medio para el caso de las laterales.

Al momento de construir la mesa debe de asegurarse que las coordenadas suministradas cumplan con las condiciones dadas en los puntos anteriores, en caso de no ser así debe notificar un error al usuario. Si se cumplen con las condiciones debe proceder a calcular las posiciones de las troneras de la mesa. Para mayor referencia puede observar la imagen 1.

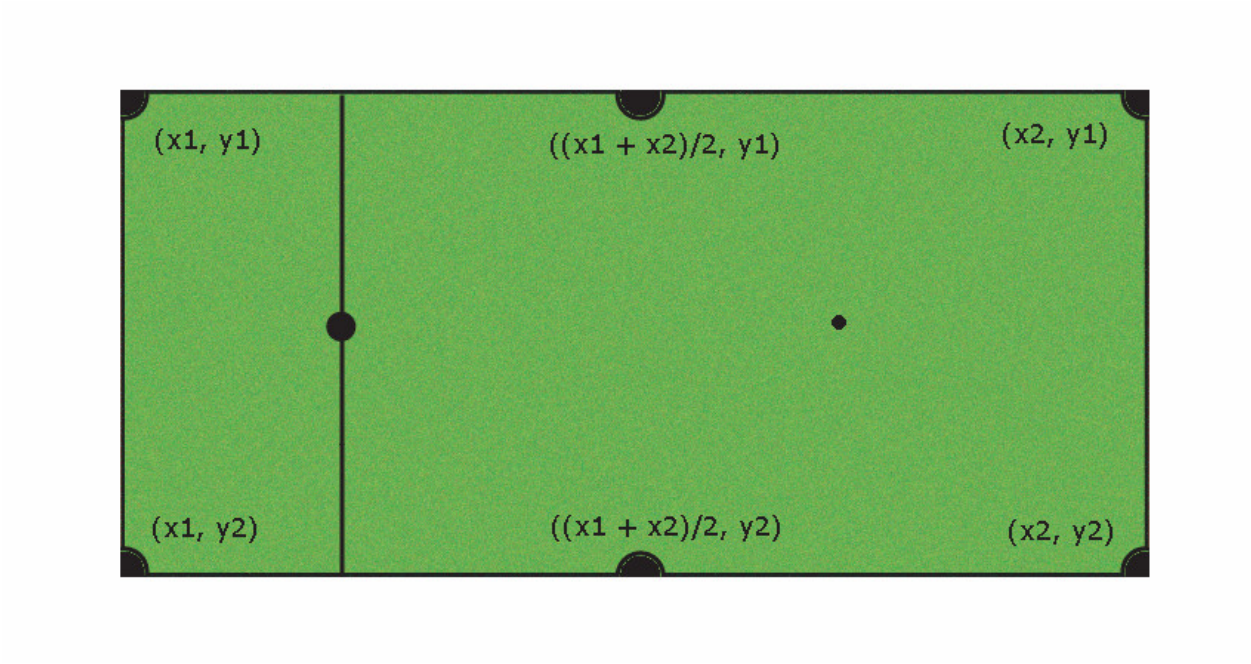

Figure 1: Representación de la mesa de pool, consiste de un rectángulo descrito por dos pares coordenados (x1, y1), (x2, y2). En la imagen se puede observar los centros de cada tronera relativos al rectángulo que describe la mesa.

2 Representación de las Bolas de Pool

Las esferas que vamos a simular se encuentran en un espacio 2D, lo que implica que para representar una bola requerimos de un par coordenado (x, y) que corresponde al centro del círculo, un radio y un número para identificarla.

Las reglas que debe cumplir una bola sobre su ubicación inicial son las siguientes:

- Debe encontrarse dentro del área definida por la mesa.

- No se puede encontrar dentro de una tronera inicialmente.
- No pueden estar solapadas entre sí.

Lo hasta ahora descrito son las condiciones necesarias para poder representar correctamente el estado inicial de la mesa, en lo que se refiere propiamente a la mesa y la configuración de las bolas de pool sobre ella.

3 Interacción Física

Al momento de realizar un tiro, se le solicita al usuario la velocidad que se le quiere aplicar a la bola blanca, dicha velocidad esta compuesta por un par coordenado (x, y), el cual define la dirección del desplazamiento y la distancia que se recorre en cada iteración de la simulación, por ejemplo, el vector $(-1, 5)$ indica que la dirección es hacia la izquierda y arriba y que la distancia que se recorre en cada iteración es una unidad en X y cinco en Y, una vez definido el movimiento pueden ocurrir los siguientes eventos:

- Se desplaza libremente sobre la mesa hasta detenerse por fricción.
- Colisiona con otra bola en la mesa.
- Colisiona con un borde de la mesa.
- Cae en una de las troneras de la mesa.

3.1 Fricción

Luego de cada iteración de la simulación, debe restar una constante al vector de velocidad de la bola que está siendo actualizada en ese momento.

Cuando la velocidad de la misma se encuentre por debajo de un umbral la misma debe ser detenida por completo.

3.2 Colisión Círculo - Círculo

Para detectar si existe una colisión entre dos círculos, debemos verificar si las mismas se intersectan de alguna manera, para calcular dicha intersección debemos verificar si la distancia entre los centros de ambos círculos es menor a la suma de sus radios, como se puede apreciar en la siguiente expresión:

$$(x_1 - x_2)^2 + (y_1 - y_2)^2 <= (radio_1 + radio_2)^2$$

donde el par (x_1, y_1) corresponde con las coordenadas del centro del círculo uno, el par (x_2, y_2) corresponde con el centro del círculo dos y $radio1$, $radio2$ es el radio de cada círculo.

Si la condición se cumple entonces los círculos se están intersectando, por tanto existe una colisión. Sabiendo esto debemos de alguna forma manejar la colisión lo cual dependerá de

con que colisionamos. Si la colisión ocurre entre una tronera y una bola simplemente se elimina la bola de la mesa ya que la misma entró en la tronera. Por otra parte si la colisión ocurrió entre dos bolas debemos calcular la reacción física entre ellas.

3.3 Manejo de colisión Círculo - Círculo

El proceso para hacer el manejo de la colisión consiste de un conjunto de pasos los cuales concluyen con una nueva velocidad para las bolas implicadas en la colisión.

El primer paso consiste en calcular la distancia mínima a la que deberían estar ambos círculos para evitar la colisión entre las mismas. Los cálculos a realizar son los siguientes:

$$Separación_X = \Delta_X * (sumRadius - |\Delta|)/|\Delta|$$
$$Separación_Y = \Delta_Y * (sumRadius - |\Delta|)/|\Delta|$$

Donde:

$$\Delta_X = X_1 - X_2$$
$$\Delta_Y = Y_1 - Y_2$$
$$|\Delta| = \sqrt[2]{\Delta_X^2 + \Delta_Y^2}$$

Luego verificamos si los círculos se encuentran viajando en direcciones en las cuales se van a separar

$$V = (velocidadX_1 - velocidadX_2) * Separación_X + (velocidadY_1 - velocidadY_2) * Separación_Y$$

Si $V > 0.0$ entonces ya los círculos se están alejando el uno del otro, por tanto no hay necesidad de continuar, en caso contrario debemos calcular la nueva velocidad para los círculos. Esta validación previa nos indicaría que ya la colisión fue manejada en una iteración previa pero que por las velocidades calculadas los círculos aún siguen intersectados.

La nueva velocidad a aplicar a cada bola se calcula de la siguiente forma

$$VelocidadBola_1 = VelocidadBola_1 + (-V)$$
$$VelocidadBola_2 = VelocidadBola_2 - (-V)$$

El manejo con las paredes de la mesa van a ser obviadas para esta fase.

4 Menú

La aplicación a desarrollar debe contar con un conjunto de menús dispuestos para guiar al usuario final durante su uso.

4.1 Menú Principal

Este menú se despliega inicialmente en la aplicación y cuenta de las siguientes opciones:

- Jugar
- Desarrollador
- Salir

En la opción desarrollador se debe desplegar los datos del programador del sistema, nombre y cédula.

La opción salir, es la única forma en la cual el usuario puede cerrar la aplicación.

La opción jugar despliega un submenú descrito a continuación.

4.2 Jugar

- Cargar Mesa
- Cargar Bolas
- Hacer Disparo
- Atras

En la opción de cargar la mesa se le debe solicitar al usuario las esquinas de la misma, realizando las validaciones pertinentes para garantizar que las esquinas suministradas cumplan con las restricciones indicadas en la sección de la mesa. Asuma que el tamaño final de la mesa será siempre lo suficientemente grande como para ubicar todas las troneras de la misma.

La opción "Cargar Bola" debe solicitarle al usuario la ubicación y numeración de cada una de las bolas en la mesa, para esta fase solo se solicitaran dos bolas, la blanca y una extra. De igual manera debe realizar todas las validaciones necesarias descritas en la sección de la "Representación de las bolas de Pool".

La opción de "Hacer Disparo", solo debe habilitarse cuando ya se haya previamente cargado la mesa y la bolas en el sistema, esta opción le solicitará una velocidad en la que quiere ser impulsada la bola blanca y una vez concluida la simulación debe de imprimir la ubicación final de cada esfera.

Todas las opciones una vez concluidas sus funciones deben regresar al menú de jugar, la opción atrás regresa al menú principal.

5 Consideraciones Finales

- El radio de las bolas de pool y las troneras es una constante con valor 1.
- La fricción es una constante que debe restársele a todas la velocidad de las bolas de pool y es 0.2.
- Toda velocidad menor a 0.1 debe ser llevada a 0.
- La velocidad máxima permitida en el sistema es 10, cualquier velocidad superior a esta producto de colisiones entre esferas, debe ser llevada a 10.
- La simulación solo se detendrá cuando se alcancen el máximo número de iteraciones, la cual es una constante con valor 360.
- La salida de la simulación debe indicar el número de la bola y su posición.
- Los mensajes de error desplegados en el sistema deben indicar el motivo del error.

Vale acotar que las velocidades que se manejaran en el sistema pueden ser positivas o negativas ya que indican dirección, esto debe ser tomado en cuenta al momento de aplicar fricción y al momento de evaluar si la bola debe ser detenida por estar por debajo del umbral o frenada por sobrepasar la velocidad máxima.

6 Reestricciones y Entrega del proyecto

- El proyecto debe ser realizado de manera individual y su fecha de entrega será hasta el Lunes 25 de Enero a las 11:59 pm.
- El mismo debe ser entregado por correo electrónico al correo francisco.morenociens@gmail.com.
- El asunto del correo debe cumplir el siguiente formato: AyP-Proyecto1-Sección-Cédula.
- En el mismo solo debe estar adjunto el archivo .cpp el cual debe estar nombrado bajo el siguiente formato Proyecto1-Seccion-NombreyApellido-Cedula.cpp sin acentos o ñ.
- Todo código enviado debe compilar correctamente.
- Las copias serán penalizadas con la nota mínima cero (0).