A ti te va a caer el Axl

Primera Entrega

Daniel Barreto - #04-36723
daniel@ac.labf.usb.ve

11 de febrero de 2009

Índice general

Introducción

"A ti te va a caer el Axl[1] *"*[2] es un juego desarrollado en 2D+1/2 que emula juegos de pelea entre un personaje principal y varios controlados artificialmente, sobre un "stage" o campo de tamaño limitado.

El objetivo principal del juego es vencer a todos los contrincantes que sean encontrados, para esto es necesario atacarlos y herirlos de forma que sean más suceptibles a ser lanzados fuera de la plataforma sobre la que pelean.

Para esta entrega se implementaron diferentes *Steering Behaviors* como persecución y evasión (*Seek, Flee, Pursue* y *Evade*), evasión de obstaculos (*Obstacle Avoidance*), comportamientos grupales o de manada (*Flocking*) que incluyen Separación, Cohesión y Alineación.

El juego esta construido en 3D usando OpenGL como herramienta para realizar el "rendering" de todos los objetos que este incluye (personajes, obstaculos, plataforma, etc).

[1] Axl Rose, vocalista principal de la banda de Heavy Metal Guns n' Roses
[2] Nombre provisional

Capítulo 1

Metáfora del Juego

La metáfora del juego se trata de la difícil tarea que tiene **Slash** (personaje principal), el guitarrista principal de la banda de heavy metal *Guns n' Roses*, de acabar con los máximos exponentes del *Reggaeton*, a los cuales debe atacar con la música producida por su guitarra.

La trama se desarrolla en una tarima de conciertos de *Guns n' Roses*, en donde se encuentran obstaculos como amplificadores, una bateria y otros instrumentos. **Slash** tendrá que sacar a los reggaetoneros invasores lanzándolos fuera de la tarima, pero este objetivo no es tan fácil como parece, pués cada enemigo viene lleno de energía y su capacidad de quedarse dentro de la tarima está ligado a su nivel de energía actual, por esto, **Slash** tendrá que lastimarlos con su guitarra antes de intentar arrojarlos hacia afuera, de forma parecida al juego de Nintendo © *Super Smash Bros.*

Capítulo 2

El juego y sus objetos

El juego consta de varios objetos que se describen a continuación

2.1. Personajes

Los personajes que participan en el juego pueden ser divididos en dos grupos: El personaje principal (**Slash**) y sus enemigos los reggaetoneros. Ambos cuentan con resticciones similares en cuanto a su máxima velocidad y aceleración, tanto lineal como angular, que puedan tener. Además cada personaje tiene una cierta cantidad de masa que lo hace más o menos suceptible a ser desplazado por golpes de otros jugadores.

Todos los personajes cuentan con un nivel de vida o energía, el cual es multiplicado por su masa cada vez que se detecta una colision de algún vector de fuerza que se le este siendo aplicado. El máximo nivel de energía para todos los personajes es de 100 puntos, y los golpes o ataques que reciban de otros personajes irán disminuyendo poco a poco dicho nivel.

La fórmula para calcular la masa de un personaje cuando reacciona frente a una colisión realizada por algún arma o bala es la siguiente:

$$M_c = M_p x E/100$$

Donde M_c es la masa referente a una colisión con un arma o proyectil lanzado por otro personaje, M_p es la verdadera masa del personaje y E es

su nivel de energía.

Para otro tipo de colisiones como los que vendrían dados por el choque con algún otro personaje, se usa la verdadera masa del personaje M_p.

2.1.1. Slash

El personaje principal del juego sólo posee un arma: su guitarra, con la cual puede golpear a los reggaetoneros como si fuera una espada, pero tambien puede disparar "balas" a sus enemigos más lejanos. Actualmente no existen restricciones a cerca del número de balas con las que puede contar **Slash**.

2.1.2. Los Reggaetoneros

En el estado en el que se encuentra el juego para esta entrega, los reggaetoneros no cuentan con ningún arma para defenderse. Su única función es demostrar comportamientos inteligentes desarrollados artificialmente. Estos comportamientos serán detallados más adelante.

2.2. Escenario o tarima

El escenario del juego es un espacio rectangular de tamaño fijo, en el cual se encuentran objetos que funcionarán como obstaculos a evitar para los personajes llevados por la inteligencia artificial.

Capítulo 3

Comportamientos y movimientos desarrollados

3.1. Los comportamientos no tan inteligentes de Slash

El comportamiento de **Slash** viene dado por la interacción del usuario con el juego, pero posee la posibilidad de realizar las acciónes "Saltar" y "Disparar".

Al saltar, se le da un valor predeterminado a la componente Y de su velocidad. En el momento del salto, la velocidad en el eje Y se ve afectada por la acción de la gravedad que actúa siendo una aceleración constante y negativa a lo largo de dicho eje.

La acción de disparar es dejada a manos del jugador. El usuario podrá elegir una fuerza y un ángulo para realizar los disparos mediante el uso del teclado. La trayectoria que siguen las balas variarán dependiendo de esas dos variables.

3.2. Los reggaetoneros

Las acciones que pueden desempeñar los reggaetoneros son las siguientes:

- **Vagar** (Wander): Los reggaetoneros pueden vagar por la tarima cada

vez que se encuentren sin ningún otro comportamiento con más importancia que tengan que realizar. Para simular este comportamiento se toma eventualmente una nueva dirección a seguir y se devuelve una fuerza o "steering" que lo acelera hacia dicha dirección.

- **Perseguir y Evadir** (Pursue and Evade): Con estos comportamientos, los reggaetoneros persiguen (o evaden) un objetivo, estimando la posición del objetivo 1 segundo mas adelante del tiempo actual, asumiendo que el objetivo permanece con la misma velocidad durante dicho tiempo.
- **Evasión de obstaculos** (Obstacle Avoidance): Los reggaetoneros tratarán de evadir, siempre que se pueda, los obstaculos que se les presenten en el camino. Entre los obstaculos a evitar, se fijarán también en los bordes del escenario, tomandolos como paredes imaginarias que tendrán que esquivar.

  Actualmente los reggaetoneros utilizan solo un rayo de visión dirigido hacia donde este dirigida su orientación, dicho rayo de visión se modela como un segmento de linea de tamaño 15.
- **Comportamientos en manada** (Flocking and Swarming): Los reggaetoneros se comportan como un enjambre o "swarm". Para poder simular este movimiento en conjunto son necesarios los siguientes comportamientos:
  - **Alineación** (Alignment): Se busca mantener una dirección de desplazamiento común para todos los integrantes del enjambre.
  - **Cohesión** (Cohesion): Los miembros del enjambre tratan de mantenerse juntos, tratando de acercarse a un centro de masa del enjambre en general.
  - **Separación** (Separation): Los miembros del enjambre tratan de mantener cierta separación entre ellos.

3.3. Comportamiento general, unión de varios comportamientos

Para lograr un comportamiento más natural en los reggaetoneros se fusionaron los comportamientos anteriormente detallados, utilizando el sistema

de “Blending” con arbitraje, en donde se agruparon los comportamientos y se establecieron prioridades entre dichos grupos.

De esta forma, cada reggaetonero se encarga de aplicar la fuerza dada por el grupo de comportamientos con mayor prioridad y que haya devuelto alguna fuerza superior a un umbral previamente establecido, que tenga entre sus comportamientos asociados.

Dentro de cada grupo de comportamientos se ejecutan los comportamientos pertenecientes a dicho grupo y se suman entre ellos, tomando en cuenta que distintos comportamientos en particular tienen un mayor peso en la suma resultante.

La implementación del “Blending” realizado será detallada posteriormente.

Capítulo 4

Detalles de Implementación

4.1. Herramientas Utilizadas y Requerimientos

El juego está siendo desarrollado con *Python* como lenguaje de programación y *OpenGL* como librería gráfica, se está usando *Debian Linux* como ambiente de desarrollo y se corre en una computadora con 512Mb de Ram, tarjeta gráfica ATI Radeon Xpress 200M de 128Mb y procesador Intel Celeron de 1.46GHz.

Se están usando varias librerías auxiliares de *Python*, entre las cuales vale la pena mencionar:

1. **Pygame:** Conjunto de módulos de *Python* diseñados para escribir juegos.
2. **Psyco:** Módulo de *Python* para agilizar y acelerar la ejecución de código escrito en este lenguaje.

4.2. Modelos Geométricos Implementados

Para facilitar el trabajo en el campo geométrico Euclideano, se realizaron varios tipos abstractos de datos que modelan estructuras importantes como:

- **Vector3:** Tipo abstracto de datos que modela un vector en 3 dimensiones. Este TAD esta implementado para funcionar como un tipo de

datos nativo de *Python*, en donde ha sido implementada la suma y multiplicación por escalares y vectores. A parte de esto, cuenta con otras operaciones importantes como producto cruz, producto punto, longitud, orientacion, etc.

Este tipo abstracto se realizó basado en el trabajo de *Will McGugan* (`http://www.willmcgugan.com`), pero se incluyeron algunas modificaciones y correcciones necesarias.

- **Rect:** Tipo abstracto de datos que modela un Rectángulo en 2 dimensiones. Es utilizado para mantener el área ocupada por los personajes del juego, incluye operaciones como: unión, intersección, detección de colisiones entre rectángulos y entre puntos con rectángulos, clipping, entre otros.

  Este tipo abstracto de datos esta basado en el Modelo Rect que viene incluido en *Pygame*, pero fue reescrito desde cero para que pudiese trabajar con coordenadas flotantes pues el modelo de *Pygame* sólo trabaja con números enteros.

Para más información sobre estas estructuras, se puede revisar la documentación del código de cada uno de ellos, donde se explica como funcionan cada uno de los métodos.

Ambos se encuentran en la carpeta `physics` del proyecto.

4.3. Blending de comportamientos

Cada comportamiento fue escrito como una función, y para dar una interfaz general a cada comportamiento se escribió la clase `Behavior` que funciona como un contenedor de argumentos necesarios para ejecutar un comportamiento como su peso y el "handler" que hace referencia a la función que debe ser ejecutada para realizar el comportamiento.

Para realizar el "Arbitraje" entre comportamientos se diseño la clase `BehaviorGroup` que funciona como un conjunto de comportamientos que tienen una misma prioridad. Dicha clase define un método `execute` que se encarga de ejecutar uno a uno todos los comportamientos que contiene y de ir haciendo "Blending" entre ellos.

Por último, los personajes manejados por la inteligencia artificial definen un método `behave` que busca todos los `BehaviorGroup`'s asociados al mismo y los ejecutan, buscando el que tenga mayor prioridad y que devuelva una fuerza o "steering" superior a cierto umbral predefinido.

Capítulo 5

Problemas encontrados

5.1. Velocidades después de una colisión

Al chocar dos personajes el intercambio de velocidades no se realiza como debería, pués todavía no se toma en cuenta la masa del personaje y se asume que su masa "no existe" (o que es igual a 1).

A parte de esto, el choque entre los reggaetoneros hace que su orientación varíe abruptamente, mostrando un comportamiento errático por un período de tiempo corto.

5.2. Problemas con la cohesión de los "boids"

Cuando los reggaetoneros actúa en conjunto como una manada, la cohesion hacia el centro de masa calculado entre ellos genera una fuerza que a veces es mucho mayor a el resto de las fuerzas requeridas en el *Flocking*. Esto genera un comportamiento inestable en el resultado final.

La solución encontrada hasta ahora ha sido darle mucho más peso a la separación en relación con el peso dado a la cohesión.

5.3. Muchas balas = Juego lento

Cuando se disparan muchas balas, el juego empieza a ponerse lento mientras estas balas están en el aire. Una vez que tocan el suelo son eliminadas de memoria y el juego vuelve a recuperar su velocidad. Esto es debido a que las balas, como son esféricas, son pesadas para hacerles "Render", y probablemente también se debe al procesador de la computadora donde se está desarrollando el juego.