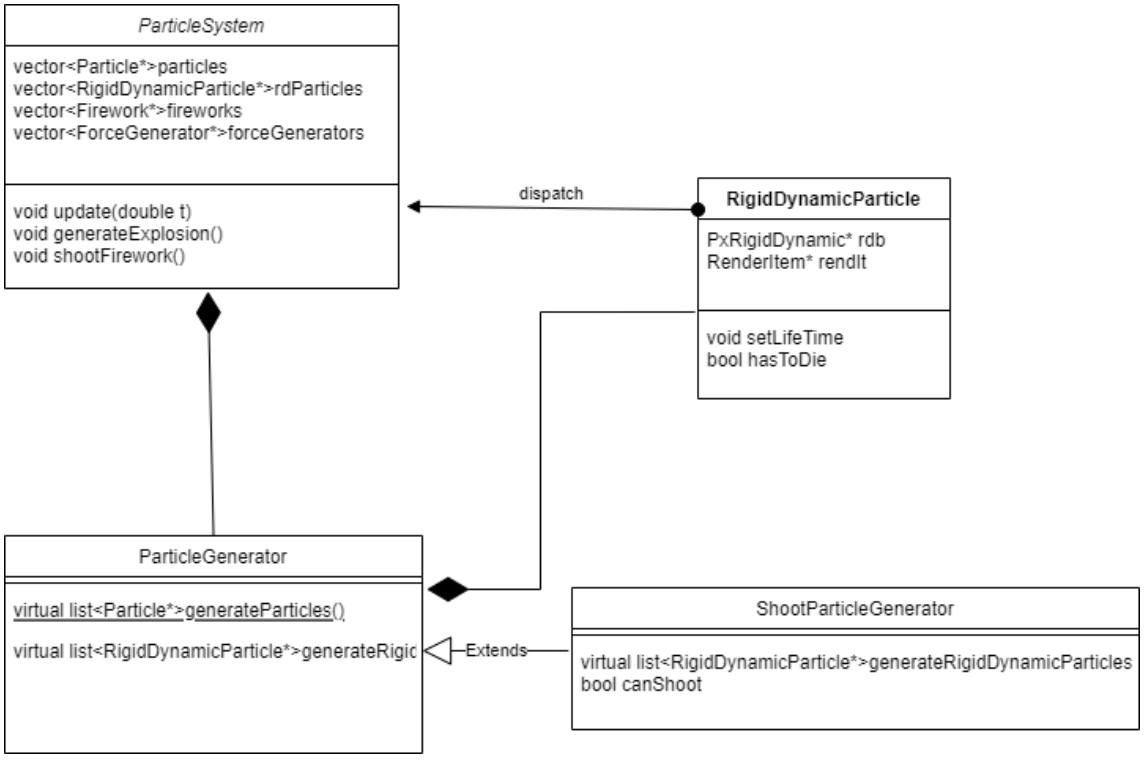
PROYECTO FINAL – SIMULACIÓN FÍSICA PARA VIDEOJUEGOS

DAVID CASIANO FLORES

Para el proyecto final de la asignatura Simulación física para videojuegos he decidido hacer una demo de lo que podrían ser las mecánicas básicas de un juego tipo shooter en primera persona y de resolución de puzles. En el juego controlamos a nuestro personaje haciendo uso de las teclas WASD para el movimiento del personaje, el ratón (pulsando click derecho o izquierdo y moviéndolo) para el movimiento de la cámara, la barra espaciadora para saltar y la tecla F para el disparo.

La escena consta de un pasillo bloqueado por un muro y una diana al final del mismo. El objetivo principal es llegar a esta última y obtener cierta puntuación. Para ello, deberemos destruir el muro que nos bloquea el camino, ya sea disparándole o activando un interruptor mediante un disparo el cual generará una explosión que destruirá el muro. Tras llegar a la diana, tendremos que acertar disparos en ella para obtener cierta puntuación y así ganar la partida. La diana se mueve de izquierda a derecha en un movimiento de vaivén (producido por un muelle al que está anclada), y además en determinados momentos aparecerán ráfagas de viento, las cuales desviarán nuestros proyectiles dificultando acertar en el blanco. Encima de la diana hay un indicador que se iluminará en rojo cuando haya viento y en verde en caso contrario. Cuando hayamos alcanzado la puntuación requerida, se dispararán unos fuegos artificiales indicando nuestra victoria.

El diagrama de clases es el siguiente:



Como podemos ver, la clase ParticleSystem se encarga de gestionar y agrupar casi todos los elementos físicos del proyecto, almacenando las partículas activas y actualizando sus propiedades después de reaccionar a los diferentes efectos físicos a los que se someten. También tenemos una clase RigidDynamicParticle, que se encarga de almacenar todo lo necesario para que haya un sólido rígido en la escena, además de que desaparezcan cuando deben hacerlo. Por último, tenemos la clase ShootParticleGenerator, que genera partículas de sólido rígido a modo de disparo.

En el proyecto tenemos dos generadores de fuerzas diferentes: el viento y la explosión. Ambos heredan de ForceGenerator. En cuanto a WindForceGenerator, para cada bala disparada se comprueba si está dentro del área de efecto del viento. Si lo está, se le aplica una fuerza al sólido rígido determinada por la siguiente fórmula:

$$\vec{F} = k_1(\vec{v}_v - \vec{v}) \qquad (1)$$

donde k_1 toma el valor 0.47 y $\vec{v}_v$ y $\vec{v}$ son las velocidades del viento y la partícula respectivamente.

En cuanto a ExplosionGenerator, tenemos un método generateExplosion el cual les aplica una fuerza a los sólidos rígidos contenidos en el área de efecto. Esta fuerza es:

$$\vec{F} = \frac{k}{r^2}\begin{bmatrix} x - x_e \\ y - y_e \\ z - z_e \end{bmatrix} e^{-\frac{t}{\tau}} \qquad (2)$$

donde k toma el valor 1000000, r es la distancia de la partícula a la explosión y τ la constante de tiempo de la explosión y toma el valor 2.

También podemos destacar algunos efectos extra que ayudan a que el jugador reciba un mejor feedback de la situación del juego. Por ejemplo, cuando una bala impacta sobre un interruptor o sobre la propia diana, la zona en la que ha impactado cambiará de color para informar al jugador.