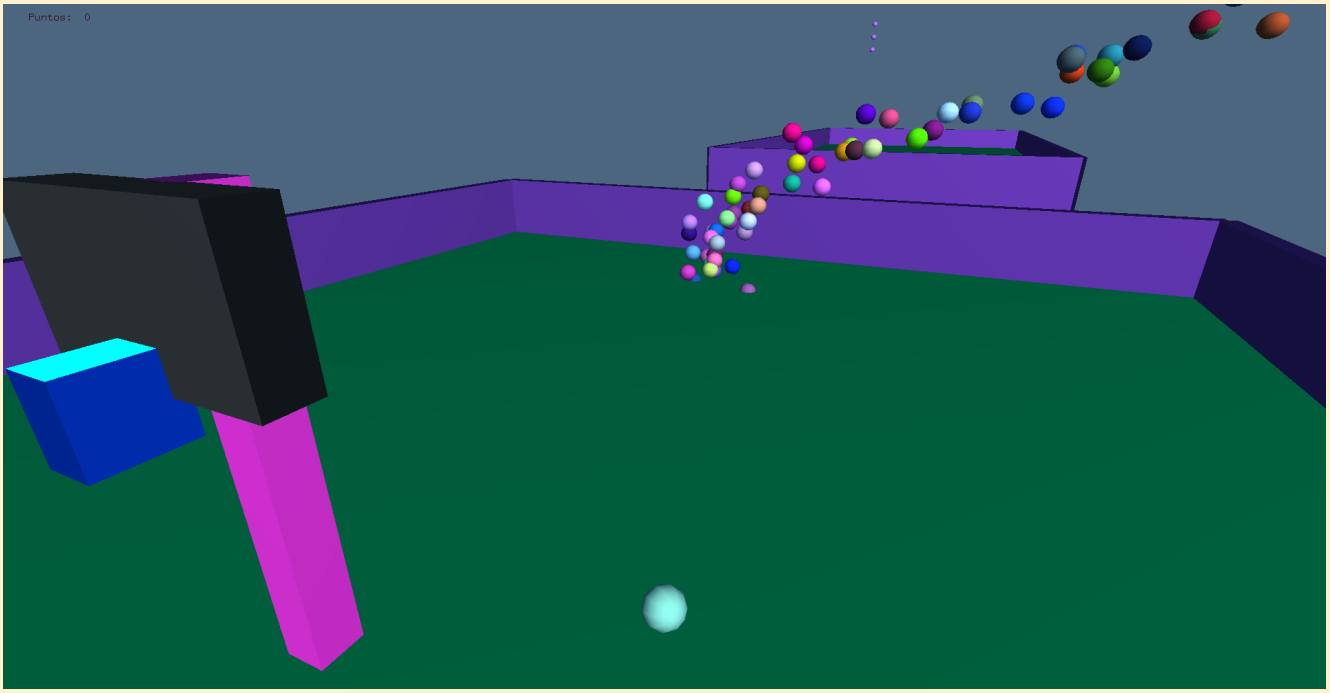
PROYECTO FINAL SIMULACIÓN FÍSICA PARA VIDEOJUEGOS
ADRIÁN MONTERO CASTRILLO

1. TEMÁTICA

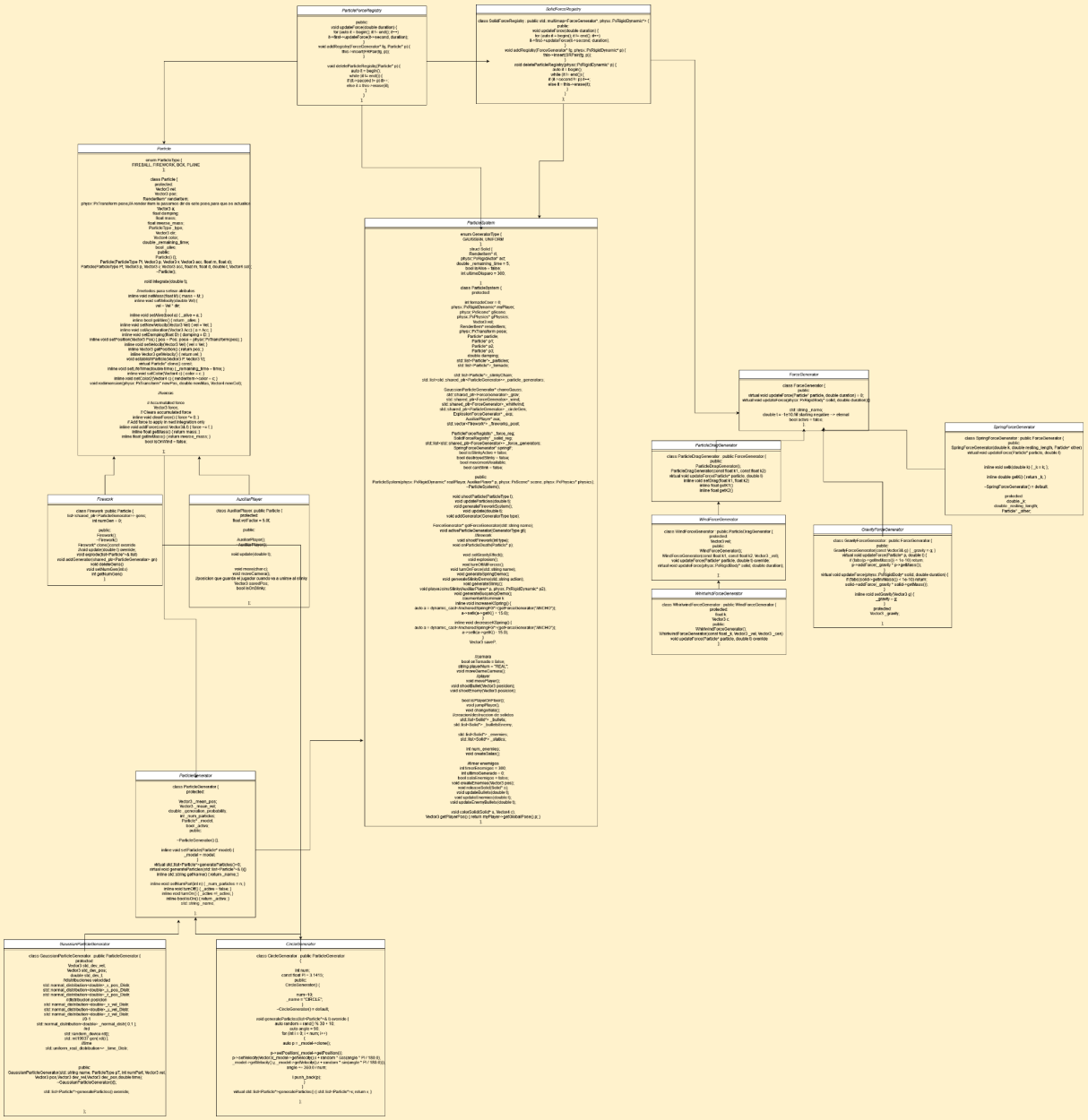


Mi proyecto final consiste en una pequeña adaptación en 3D del juego de móvil Bounce, desarrollado por Nokia (https://es.wikipedia.org/wiki/Bounce_(serie_de_videojuegos))

El proyecto está realizado en la librería Physx de Nvidia. En el juego manejaremos a una pequeña pelota, que podrá moverse a su gusto por dos salas, ya sea rodando por el suelo o dando saltos.

El objetivo es ir investigando un poco diversos efectos físicos, tirando a canasta, dando saltos, matando enemigos,etc.

2. DIAGRAMA DE CLASES (disponible en Github)



3. ECUACIONES FÍSICAS

A continuación, procedo a explicar las ecuaciones físicas que he empleado en la realización del proyecto.

class WindForceGenerator:

$\vec{Fv}=k1(vv-v)+k2 \parallel vv-v\parallel(vv-v)$

El generador de fuerza de viento es utilizado al disparar a los sólidos rígidos enemigos, aplicándose a ellos y saliendo volando por los aires. La constante k1 recibe un valor de 0.6 y a los sólidos rígidos afectados se les aplica una velocidad = Vector3(20,0,0)

```
void WindForceGenerator::updateForce(physx::PxRigidBody* solid, double duration)
{
    if (fabs(solid->getInvMass()) < 1e-10) return;

    Vector3 v = vel - solid->getLinearVelocity();
    float drag_coef = v.normalize();
    Vector3 dragF;
    drag_coef = _k1 * drag_coef + _k2 * drag_coef * drag_coef;
    dragF = -v * drag_coef;

    solid->addForce(dragF);
}
```

class WhirlwindForceGenerator: $vtorbellino(x,y)=K[-(z-zc),50-(y-yc),x-xc]$

El generador de fuerza de torbellino es aplicado a las partículas generadas por el generador gaussiano, simulando un "túnel de teletransporte". Al sobrepasar el jugador la posición de dicho torbellino, al jugador se le aplicará dicha fuerza, y al cabo de un tiempo se teletransportará a la segunda sala. La constante K recibe un valor de 1.9, la velocidad Vector3(1,0,0) y el centro se encuentra en Vector3(0,0,0).

```
void updateForce(Particle* particle, double t) override {
    if (fabs(particle->getInvMass()) < 1e-10) return;

    auto p = particle->getPosition();
    vel = { k * (-(p.z - c.z)),k * (35 - (p.y - c.y)),k * (p.x - c.x) };

    Vector3 v = particle->getVelocity() - vel;
    float drag_coef = v.normalize();
    Vector3 dragF;
    drag_coef = _k1 * drag_coef + _k2 * drag_coef * drag_coef;
    dragF = -v * drag_coef;
    particle->addForce(dragF);

}
```

class GravityForceGenerator:
Las partículas a las que se les aplique esta fuerza estarán sometidas a gravedad terrestre (-9.81m/s^2)

```
virtual void updateForce(Particle* p, double t) {
    if (fabs(p->getInvMass()) < 1e-10) return;
    p->addForce(_gravity * p->getMass());
}
```

Cámara:
La cámara seguirá en todo momento al jugador, podremos girarla y cambiar la dirección a nuestro gusto, sobre todo para disparar ya que las balas salen según dicha dirección.

```
void ParticleSystem::moveGameCamera()
{
    Vector3 cameraPos;
    if (playerNum == "REAL") {
        cameraPos = myPlayer->getGlobalPose().p;
    }
    else  cameraPos = _player->getPosition();

    auto auxDir = GetCamera()->getDir();
    auxDir.y = 0;

    cameraPos.y += 20;

    cameraPos -= auxDir * 20;
    GetCamera()->mEye = cameraPos;
}
```

Colisiones:
Las colisiones del juego se dividen en dos partes. Las siguientes colisiones son comprobadas mediante nombres, que comprueban si hemos metido canasta o si nos ha dado una bala enemiga, actualizando los puntos.

```
void onCollision(physx::PxActor* actor1, physx::PxActor* actor2)
{
    if (actor1->getName() == "Bullet" && actor2->getName() == "Canasta") {
        particleSys->shootFirework(0);
        puntos += 1;
    }
    if (actor1->getName() == "BulletEnemy" && actor2->getName() == "MyPlayer") {
        puntos -= 1;
    }
}
```

Por otra parte, se comprueba si han colisionado nuestras balas con los enemigos.

```
//Metodo para actualizar solidos
void ParticleSystem::updateBullets(double t)
{
    auto s = _bullets.begin();
    while (s != _bullets.end()) {
        if ((*s)->_alive) {
            for (auto e : _enemies) {

                if (physx::PxGeometryQuery::overlap((*s)->rI->shape->getGeometry().sphere(), (*s)->act->getGlobalPose(), e->rI->shape->getGeometry().box(), e->act->getGlobalPose())) {
                    auto e2 = static_cast<physx::PxRigidDynamic*>(e->act);
                    _solid_reg->addRegistry(getForceGenerator("WIND"), e2);
                    colorSolid(e, { 1.0,0.0,0.0,1.0 });
                }
            }
            (*s)->_remaining_time -= t;
            if ((*s)->_remaining_time <= 0) {
                releaseSolid(*s);
                s = _bullets.erase(s);
            }
            else
                s++;

        }
    }
}
```

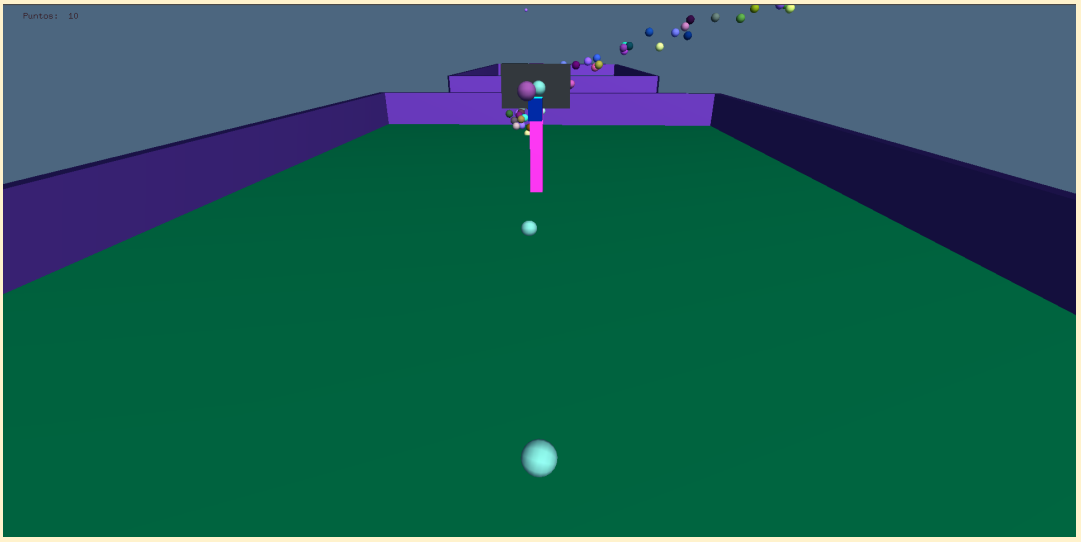
4. EFECTOS

- Disparar balas, ya sea para meter canasta o para matar enemigos
- Cámara que sigue al jugador
- Emisión de fuegos artificiales al anotar canasta
- Torbellino de partículas en el que se puede meter el jugador
- Fuerza de viento y cambio de color aplicados a los enemigos al ser disparados
- Contador de puntos en la parte superior izquierda, que se actualiza al meter canasta y ser disparado por los enemigos
- Jugador que salta, se mueve y dispara mediante el teclado
- El jugador puede unirse a una cadena de partículas simulando un muelle slinky

5. MANUAL DE USUARIO

Para mover el ángulo de la cámara haremos click izquierdo y moveremos el ratón. El jugador se mueve únicamente con la tecla W, desplazándose según la dirección de la cámara. Además, el jugador puede saltar pulsando la tecla J.

En el juego disponemos de dos salas, al iniciar nos encontramos en la primera, en la que encontramos una canasta, en la que tenemos que intentar anotar disparando con la tecla X, si metemos, saldrán fuegos artificiales y explotarán en el cielo.



Si avanzamos al tornado que hay en la parte del fondo de la sala, nos meteremos en él y avanzaremos a la segunda sala.

En esta segunda y última sala, si pulsamos la tecla S, nos unirá a una cadena de partículas muelle y se nos aplicará la correspondiente fuerza, al pulsar de nuevo la tecla S se dejará de aplicar dicha fuerza y comenzarán a aparecer sólidos rígidos, los cuales nos dispararán, y si nos impactan nos quitarán puntos, podemos dispararlos también con la tecla X, y nos sumarán puntos, al igual que cuando metemos canasta.

Además, si les damos con una bala, dichos solidos se volverán de color rojo y se les aplicará una fuerza de viento.

Solo podemos disparar las balas si no existen más de tres en ese momento en la escena.