CS3127 Problem Set 4
Design explanation

Angad Singh
U099118R

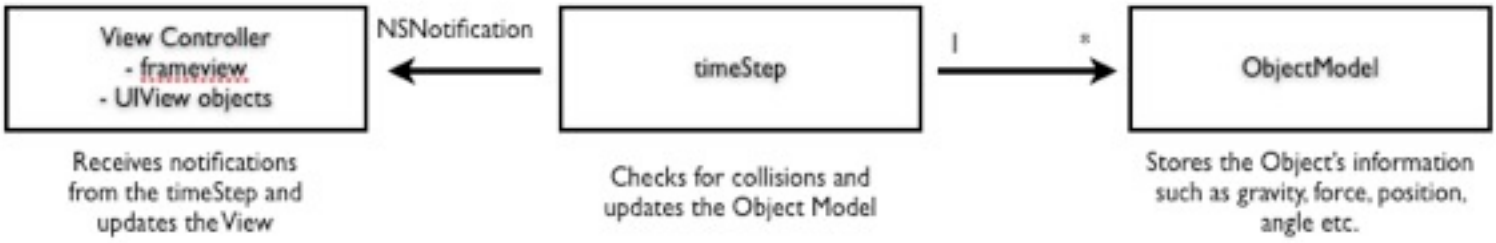
Data model

PhysxViewController
- UIView framearea
- UIView rectangles ...
- UIView walls ...

timeStep
- ObjectModel rectangles ...
- ObjectModel walls ...
- NSArray allObjects
-NSArray walls

ObjectModel
enum ObjectType
double mass;
double momentOfInertia;
Vector2D *position;
double width;
double height;
Vector2D *velocity;
double angularVelocity;
ObjectType objType;
double rotation;
Matrix2D *rotationM;
CGPoint center;
Vector2D *gray;
BOOL collided;

Module Dependency Diagram



Explanation

The ViewController sets up the accelerometer and delegates the event to the timeStep class. It then sets up the basic view - with all the rectangles and the walls. It receives notifications from the timeStep class and accordingly moves the rectangles.

The timeStep function implements the accelerometer event handler. For every accelerometer event, it applies gravity to the rectangles and checks collisions of objects with one another.
The timeStep class' constructor initializes the objectModels for different objects on screen.

The objectModel class stores the object information in various variables as listed above. It also implements the collision testing for 2 objects.

Rationale - Other alternative was to have both the timeStep and PhysxViewController in one class. But that would have made the design messier and it would not be inline with the MVC pattern.
Also, the collision detection could have been inside the timeStep class, instead of the ObjectModel class.

Extend for more complex shapes - The Physics engine can be extended to complex shapes with the ObjectModel's ability to identify which objects are we colliding. If its a circle collision, the collision detection function detects the type and automatically uses the algorithm for circle (which is actually pretty easy - you just need to see if (radius1 + radius2) < (center1-center2)).

Also if its a complex polygon, then I had used an algorithm called the PNPOLY algorithm in the collision detection problem set. The same can be used to detect collisions in the physics engine for complex shapes.

The rest of the stuff remains pretty much the same.

Testing
Black Box testing
- Gravity
    - True. Objects fall down with increasing velocity.
- Two object collision
    - True. Objects collide and have the correct impulse.
- Three object collision
    - True. Objects collide and have the correct impulse.
- Friction.
    - True. Object when held agains a wall and given a perpendicular gravity, falls with reduced velocity due to friction.
- Rotation
    - True. Object rotates with correct angularVelocity