Gizmoball: Physics Loop

CS308 Group MW1

The physics loop is the part of the project responsible for detailing how physical interactions in GizmoBall should be handled. In particular, how the ball(s) and the several Gizmos interact across the Board. The physics loop runs at every tick, and the calculations performed in it make use of the formulas defined in the MIT physics package that has been provided, which will be untouched during the course of the project. This document explains how ball motion, collision detection, collision resolution, triggering, friction/gravity, and drawing are handled in the loop.

The physics loop includes several stages that run in the following order:

1. Application of friction and gravity
2. Ball motion
3. Collision detection (if no collision is detected, jump to Stage 6)
4. Collision resolution (if no triggering occurs, jump to Stage 6)
5. Triggering
6. Redrawing

Stage 1: Application of friction and gravity

The first stage of the loop is the application of friction and gravity to the ball. The application of both must happen either at the beginning or at the end of the loop, but never in between Stage 2 and 6. A decision was made by the group to apply the external forces as the first stage of the physics loop. Since the timeUntilCollision() methods from the physics.Geometry class assume constant ball velocity for the time until the next collision to be calculated, which means the ball follows a straight-line path, Gravity and Friction are applied at each tick, in order for timeUntilCollision() to take both into account at each loop. The physics loop then enters Stage 2.

Stage 2: Ball Motion

The ball moves according to the set ball velocity and the external forces applied beforehand. There is only one move per tick time.
The loop then enters Stage 3.

Stage 3: Collision Detection

For each ball movement the timeUntilCollision() methods are used to calculate the exact amount of time it would take for the ball to collide with each of the Gizmos on the Board or other balls (if there is more than one on the Board), based on the current trajectory of the ball. This is done by mathematically determining the shapes (composed by a combination of circles and line segments obtained from the classes physics.Circle and physics.LineSegment respectively) that make up each Gizmo.

The minimum of all these times, called mintime, is then calculated. Mintime will be the time of the next collision and is also calculated for all Gizmos (or other balls) on the Board, given the current ball trajectory.
The next stage starts depending on whether a collision occurs. In mathematical terms if the minimum time to the next collision is greater than the tick time there is no collision taking place. The position of the ball(s) and the Gizmos are then updated based on their current velocity and angular velocity and the physics loop jumps straight to Stage 6.
However if the minimum time to the next collision is less than (or equal to) the time to the next collision a collision occurs and the physics loop enters Stage 4.

Stage 4: Collision Resolution

After the client updates the positions of the balls and Gizmos to account for minimum time to the next collision passing, as stated towards the end of Stage 3, the ball and the Gizmo (or other ball) about to be hit are exactly adjacent to one another.
The client then updates the ball velocity based on the collision by calling the appropriate reflect() method in the physics.Geometry class to calculate the change in the ball's velocity. Since during a collision between a ball and a Flipper(s) (or with another ball) both objects are in motion the methods used to detect the collision and change the ball velocity and trajectory will be different from the ones used in the case of a normal collision (ball vs static Gizmo).
If the collision happens between a ball and a connected Gizmo the physics loop then enters Stage 5, otherwise it jumps to Stage 6.

Stage 5: Triggering

The Triggering System is called when a collision occurs between a ball and a Gizmo that is connected to one (or more) Flipper(s). The connected Flipper(s) then activate and move at every tick.
More information on how it works is found in the Triggering System document.
The physics loop then enters Stage 6.

Stage 6: Redrawing

In order for the Board to be updated and keep displaying the the latest data obtained from the Model that concerns any type of motion on the Board, a decision was made to redraw the entire Board at every loop. This way there is no risk of not being able to notice high quality graphical collisions.
As stated in Stage 2 only one move occurs per tick time, in order to prevent having multiple collisions within each tick time, which would be troublesome for the redrawing stage.
All of the observers are notified using the notifyObservers() method, the update() method in the view package is called, and the updated view is redrawn.
The physics loop terminates with this stage and eventually starts again from Stage 1.