**Description of physics loop**

The ball movement simulated by updating the ball coordinates at each constant interval (that was selected as 20 times per second) and redrawing the board to reflect the ball’s position. At each *tick interval*, the following happens:

1. collision detection;
2. collision resolution/unhindered motion;
3. triggering if collision occurs;
4. effect of gravity on the ball;
5. effect of friction on the ball;
6. repainting;

Collision detection and handling mechanisms are independent from the effects of forces (gravity and friction) on the ball. Gravity and friction are factored in before the collision sequence to make velocity consistent during the collision detection and handling.

**Collision Detection**

Collision detection mechanism relies on calculating the *time until collision* by iteratively comparing the times to collide with all of the existing segments on the plane and selecting the minimum value (*mintime*). The *mintime* is compared to the tick interval to determine whether or not a collision will occur within that tick interval and therefore calculate the appropriate displacement value.

Depending on the type of object, collision detection relies on invocation of the following methods of Geometry class from MIT physics package: *timeUntilWallCollision, timeUntilCircleCollision, timeUntilRotatingWallCollision, timeUntilRotatingCircleCollision, timeUntilBallBallCollision*.

**Unhindered motion**

If the mintime is greater than the tick interval, then no collision occurs within the tick interval and the new coordinates of the ball are calculated by adding the displacement value (s) to the current coordinates of the ball, where time (t) is the tick interval times the ball’s velocity (v);

*s = t \* v* (1)

**Collision Resolution**

If the minimum time is less than the tick interval, there is an impending collision. Collision induces the change in the ball’s velocity vector that needs to be accounted for by implementing the following steps:

1. the ball’s position is adjusted to be adjacent to the surface of the object it is about to collide with by applying the formula (1), where time (t) is *time until collision* derived in step 1);
2. the velocity vector of the ball is updated to emulate the change in direction (by inverting either x or y components) and change in speed (by changing the magnitude of x or/and y components depending on an appropriate coefficient of reflection);

The calculations for the velocity vector are calculated by the following methods of Geometry class from MIT physics package: *applyReflectionCoeff, reflectWall, reflectCircle, reflectRotatingWall, reflectRotatingCircle, reflectBalls*.

**Triggering**

At the moment when the displacement is applied for the ball to be adjacent to the other object, a trigger is invoked.

**Gravity**

Gravity increments the y-component of the velocity vector at each tick interval (∆t) to imitate constant acceleration. Taken that the gravity is 25L/sec2 as per specifications, such displacement corresponds to 1.25L per interval squared. Hence, the displacement in y direction is increased by a factor of 1.25L during each interval. Considering that the size of the plane is 20L by 20L and L = 25px, the 1.25L transforms into the y-displacement (∆v) of 31.25px per ∆t.

**Friction**

Friction decrements the velocity vector of the moving ball to imitate the resistive force of the surface. Because the ball stays on the surface at all times, the friction applies in the counter direction of the ball motion. Friction takes into account the velocity of the moving ball and well as the change in time (∆t). The velocity of the moving ball is scaled using the two frictional constants µ and µ1 with the values of 0.025 per second and 0.025 per L respectively in the equation (2). The formula decreases the magnitude of velocity of moving ball in the directions of x and y axes.

*Vnew = Vold \* (1 – µ\*∆t - µ1\* |Vold| \* ∆t)* (2)

**Drawing**

The user interface is redrawn to reflect the change in position of the ball by notifying the Observer after friction and gravity forces are applied and a collision is detected and handled.