2D ANIMATION OF MULTIPLE COLLIDING DISCS

MATT WEBER AND JOONHO BAEK

Abstract. There are multiple techniques for implementing collision detection to support 2D animation of discs. We investigate two such approaches, Periodic Interference Test, in which collisions are checked at each frame, and Predicted Instant of Collision, in which the time of future collisions are calculated. A technique for animating the elastic shock of the colliding discs is also presented. Finally, we evaluate the performance of both and present our results.

1. Disc Collision

Animating discs colliding with each other can be achieved by predicting the impending collisions at each frame and bouncing the discs off of one another when such a collision is about to occur. Here we give the background of interference detection, collision prediction and elastic shock. Then we explain how they are used in an application.

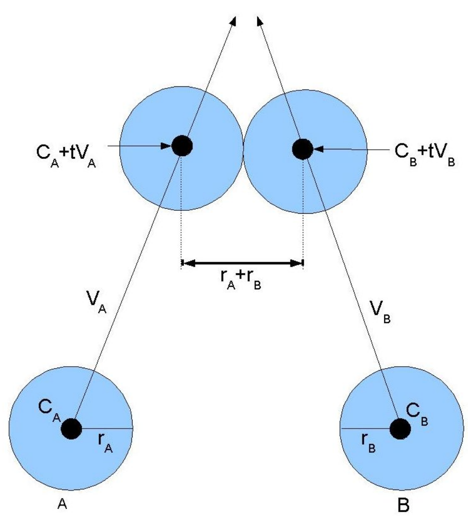


Figure 1. Calculating the Collision Time of Discs Moving at Constant Velocity

1.1. **Interference Detection.** A circle can easily be represented by a center point C and a radius r. All the points P along the boundary of the circle can then be constrained to those where their distance to C of the circle is equal to r, or more formally $\{P : \|PC\| = r\}$. Intuitively, two circles A and B touch if the distance between their centers are equal to the sum of their radii, or $\|C_AC_B\| == r_A + r_B$. It follows that if $\|C_AC_B\| < r_A + r_B$ then the circles intersect and if $\|C_AC_B\| > r_A + r_B$ then they are disjoint.

1.2. **Collision Prediction.** A point P moving across a plane with constant velocity V can be expressed parametrically as P(t) = O + tV. In the case of a circle moving at a constant velocity its center can be expressed similarly as P(t) = C + tV. So to calculate the time that two circles moving toward each other will touch we just need to find the value for t when the centers of

Date: September 30, 2008.

the circles satisfy the conditition mentioned above $\|C_A C_B\| == r_A + r_B$. To get an equation that can do this we simply plug the parametric equation into our condition and we get $(C_B + tV_B) - (C_A + tV_A) = r_A + r_B$, keeping in mind that $C_A = P_A(0)$ and $C_B = P_B(0)$ or are the origin of the circles. By squaring both sides and solving for t we get $t = \frac{(-(2C_A C_B \bullet (V_B - V_A)) \pm \sqrt{(2C_A C_B \bullet (V_B - V_A)t)^2 - 4((V_B - V_A)^2)(C_A C_B^2 - (r_A + r_B)^2)}}{2(V_B - V_A)^2}$. The lower of the values for t is the next time of collision.

1.3. **Elastic Shock.** An elastic shock assumes that there is no energy loss during collision. To have more intuitive explanation, let's assume that two objects have the same mass. At the time of collision, we can draw the tangential line at the collision contact point. Let N be the unit vector that's normal to that tangential line, $N = \frac{C_A C_B}{\|C_A C_B\|}$. The collision only affects the velocity component of each moving objects that is parallel to N, which are $U_A = (V_A \bullet N)$ and $U_B = (V_B \bullet N)$ and the elastic shock forces two objects with the same mass to exchange their velocities parallel to N, which leads to new velocity $V'_A = V_A - U_A + U_B$ and $V'_B = V_B - U_B + U_A$.

1.4. **Practical Application.** These concepts can be applied to animate discs colliding into, and bouncing off of, one another. By defining each disc by its center point and its radius, using each frame as the the unit for time, and defining velocity as the displacement vector of the discs, the values can be plugged into the equations above to determine their updated location through time. At each frame the collision time from each disc to each other is calculated, if the time is between 0 and 1 this means that the discs will collide before the next frame and their new locations should reflect the elastic shock described above. If the predicted time is greater than 1 then the discs should stay on their current path.[1]

2. Implementation Technique Comparison

As we now know the algorithm to detect the interference, predict the collision time and calculate the new velocity after the elastic shock, here we introduce implementation techniques to calculate the collision detection.

2.1. **Periodic Interference Test.** PIT (Periodic Interference Test) checks which pairs of disks interfere at each frame and for these pairs, computes new velocities. It's very coarse-grained algorithm because it checks the collision detection only at the frame rate.

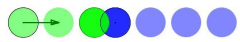
2.2. **Predicted Instant of Collision.** PIC (Predicted Instant of Collision) computes the time t of first collision (if one occurs before the next frame), advances the animation to t, computes new velocities after the shock, and then does it again. If no collision occurs before the next frame, it advances the balls to the next frame. Compared to PIT, PIC is much more fine-grained. However,it has its limit; it's hard to predict the collision time if the trajectory of the moving object is not linear with time t.

3. Experiment

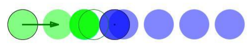
Here we show 4 different cases where PIT is wrong and explain why PIT doesn't work correctly with each case.

- Case 1: One disk is delayed.

PIT



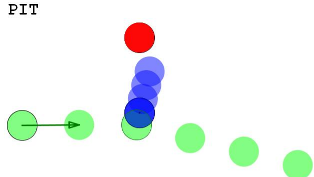
PIC



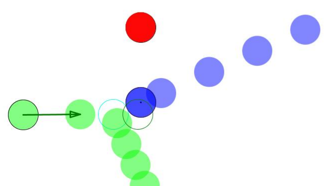
PIT detects the collision event later than the time when actual collision happens. As the result, the disk move at PIT is delayed compared to PIC which computes the collision time correctly.

- Case 2: wrong velocities after the shock

PIT



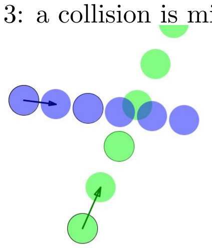
PIC



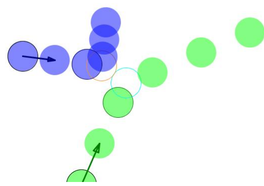
PIT detects the collision later than the actual collision time and computes the wrong velocities. As derived in section 1.3, the new velocity calculation after the elastic shock depends on $C_A C_B$ vector and the later detection of the collision with PIT skews $C_A C_B$ leading to the wrong velocity calculation.

- Case 3: a collision is missed.

PIT



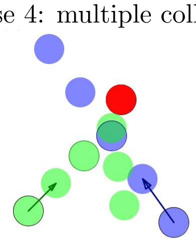
PIC



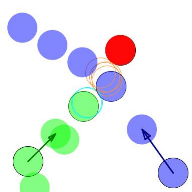
Here PIT doesn't even recognize the collision. The collision happens between third and fourth frame, but at fourth frame, PIT doesn't detect the interference between two disks as two disks passes each other.

- Case 4: multiple collisions are not handled properly.

PIT



PIC



Here PIT detects only two collisions whereas PIC detects 4 collisions correctly. This also happens because PIT doesn't detect the collision at the correct moment and even when it detects the collision, it doesn't elicit the correct velocity after the shock.

4. Conclusion

As shown in section 3, PIT suffers from its limit of synching its calculation with the frame rate. But if the animated object moves slowly compared to the frame rate, PIT will give us reasonably good answer. Compared to PIT, PIC can detect the collisions accurately as long as we can predict the collision time correctly. We assume at section 1.2 that the disk moves at the constant speed in a linear fashion, so the collision time prediction is relatively easy. However, if the motion of the object moves nonlinearly with time like a circular motion, calculating the collision prediction time accurately would be much more difficult as the equation gets more complicated.

References

1. Jarek Rossignac, *Cs6491: Computer graphics*.