Physics Final Project

Marcos Ramirez Aceves
40284094@live.napier.ac.uk
Edinburgh Napier University - Physics-Based Animation (SET09119)

Abstract

This report will focus on two parts of the final physics project. The first one is about scalability with collision detection, which basically consist on trying to make use of the maximum number of balls while calculating ball to ball and ball to cushions collisions and maintaining a smooth simulation at a frame rate of 60 fps. The second part consist in applying more complex behaviours like spin transfer between balls or sliding vs rolling.

Keywords – Physics, Collisions, Pool, Rotation, Spin, Cushions

1 Scalability

The first approach for calculating the collisions between the balls in the pool table was checking each ball at a time, this means that when performing ball to ball collisions was necessary to implement two loops that went through all the balls, when dealing with an small amount of balls this was enough, however, as in this part the aim was to maintain the maximum amount of balls possible in the simulation running smoothly, another method was needed. The approach chosen was Grid-based partitioning.

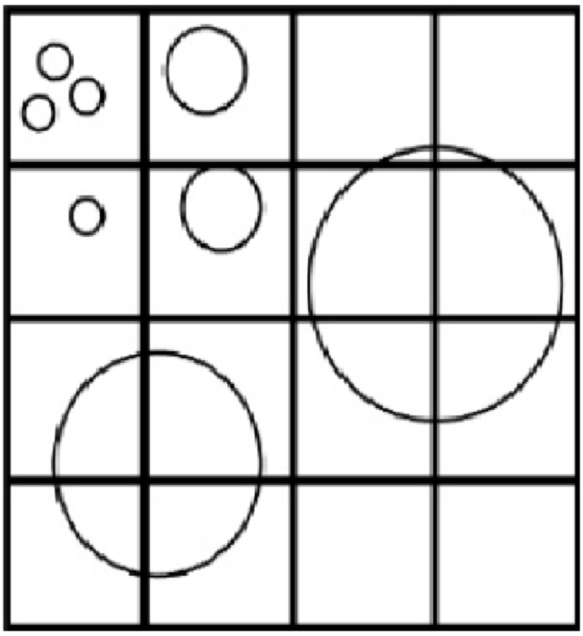

Figure 1: **Uniform Grid**

This method basically consist on creating an uniform grid based on the dimensions of the table, the grid itself consist in a 2D array of vectors which corresponds to having a vector for each of the cells of the grid. With this method the collisions are implemented by going through each cell of the grid instead of going through each ball twice.

The main challenge of this approach was finding an appropiate cell size as it can affect the performance. Apart from that, further optimization relied on two elements. The first one consisted on checking the collisions between the balls and the cushions of the table when the row/column was the first/last as only on those situations the cushion collisions were possible. And the second one consisted on only checking for collision between balls when there were more than one sphere in the cell's vector.

With this approach the maximum number of spheres with the simulation running at 60 fps achieved was of 9000 spheres.

2 Pool Physics

The second part that is going to be discussed in this report consist in the implementation of more complex behaviours in the simulation.

2.1 Sliding vs Rolling

The first element researched was when a ball stops sliding and starts rolling. In pool if a ball is struck with the cue stick at its center of mass is not going to rotate initially, instead, is going to slide through the table. However, at some point the ball stops sliding and starts rolling, this is caused because of the friction between the ball and the table.

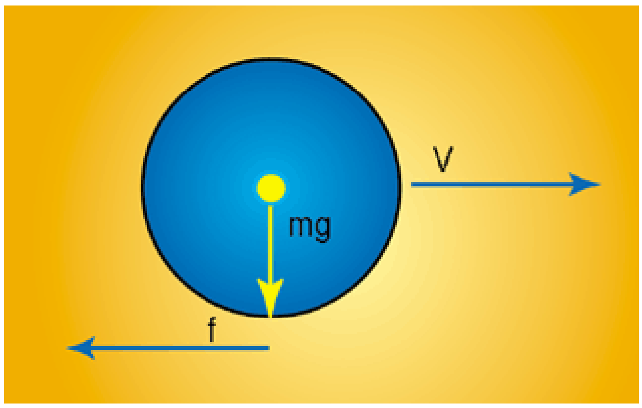

Figure 2: **Friction Force**

The force of friction applied to a body that slides over a surface is given by the following formula:

$f = \mu_s mg$

This friction force is applied in the opposite direction of the linear velocity of the ball and is applied at the surface of the ball not at the centre of mass. This friction force is going to increase the angular velocity of the ball until an equilibrium is reached. This equilibrium happens when the velocity of the point contacting the table equals to the velocity of the centre of mass of the ball, at this moment the ball start rolling. In mathematics this is expressed with:

$v = R\omega$

The approach followed to implement this in the simulation was to use a torque during the integration for the rotation. A torque is expressed as the cross product between r and f, r being the vector from the centre of mass to the point of contact of the sphere with the table and f being the friction force expressed as the following:

$f = \delta\mu_s mg$

In comparison with the previous formula here δ is added, which is just the direction of the friction force, the opposite direction of the linear velocity of the sphere.

However, as explained above, when an equilibrium between the velocity of the centre of mass and the point contacting the table is reached the torque should no longer be applied. The problem with this part was that measuring the exact moment when that happens is quite complex. To solve that, the torque was no longer applied when:

$v <= R\omega$

2.2 Spin with table cushions

The next element researched was how ball-cushion collisions affect the spin. From the papers found online was observed that all of the methods for ball-cushion collisions used a wide range of parameters including the angle previous and after the collision, the angular velocity in the three different axis or the velocities previous and after the collision. Because of its complexity and the time left, another approach was taken into account.

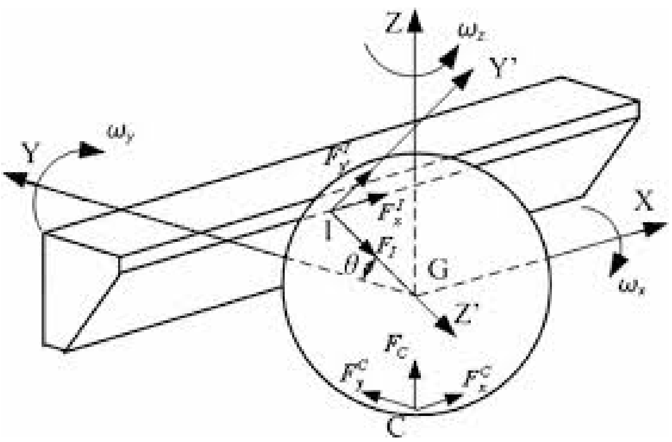

Figure 3: **Ball-Cushion Collision**

The approach mentioned above, basically consisted in a trial and error searching for a realistic behaviour. One of the elements taking into account was that the angular and linear velocity, both affected how the spin was going to be after the collision. Therefore based on those velocities the spin of the ball was either reduced or changed to go in the opposite direction.

This is obviously not a physically correct approach, however it provided decent realistic results.

2.3 Spin transfer between pool balls

The final element researched was spin transfer in ball to ball collisions. In this case, the research was done in two parts, the first one consisted in looking at how the different types of spin affected the ball collisions in slow motion videos and the second one consisted in finding papers that showed the equations needed to implement such behaviour. However, the same problem as in the last section was present in this part, the approach shown by the papers is a bit complex for the time provided and therefore another one was needed.

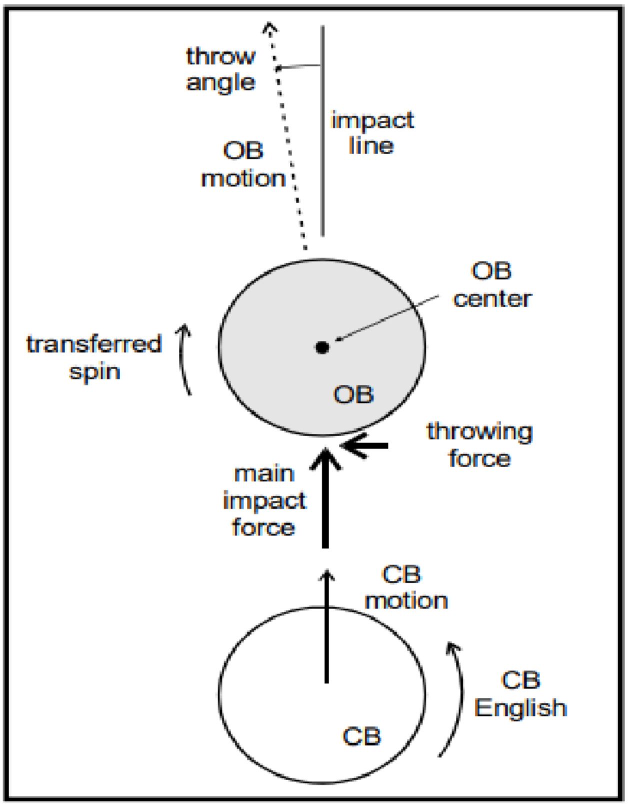

Figure 4: **Ball-Ball Spin Transfer**

For the implementation of the spin transfer in the simulation, the basic idea of transferring the spin from one sphere to the other in the opposite direction was used. Basically when applying the impulse to change the angular velocity, that impulse was multiplied by the opposite direction of the angular velocity of the sphere that was colliding with it.

3 Conclusion

In relation with the scalability part, with the model used (fixed size grid) and with the different optimisations implemented the results are quite good. Further improvement could be done by using a different model like an Octree which is a Tree-based decomposition model.

With respect to the second part, the most physically accurate element implemented was the friction force. The other two approaches were just approximations of behaviours of the balls in the pool table. That is the reason why, in the simulation, in general the balls spin as expected but sometimes some of the balls rotate in a strange way. For this part, further optimization will require to change the way in which the collision affects the spin by using more correct physics models and also consider the effects of the spin in the trajectory of the balls.

References

[1] Pool Game: https://graphics.stanford.edu/courses/cs248-videogame-competition/cs248-01/magic_README.txt

[2] Friction Force Implementation: http://www.darwin3d.com/gamedev/articles/col0999.pdf

[3] Friction Force Implementation: https://www.real-world-physics-problems.com/physics-of-billiards.html

[4] Friction Force Implementation Code Example: https://github.com/dwilliamson/GDMagArchive/tree/master/sep99/lander/Code/OGL/Pool

[5] Types of Collisions: https://www.youtube.com/watch?v=rask8MYJaB4

[6] Ball-Cushion Collisions: https://billiards.colostate.edu/physics/Mathavan_IMechE_2010.pdf

[7] Ball-Cushion Collisions: https://www.real-world-physics-problems.com/support-files/3d_ball_bounce.pdf

[8] Ball-Ball Spin Transfer: https://billiards.colostate.edu/bd_articles/2007/march07.pdf

[9] Ball-Ball Spin Transfer: https://pdfs.semanticscholar.org/e15d/959fd4c78faab433864abac3baf5625a9ae4.pdf