Lead

|  |  |
| --- | --- |
| Radius | 0.02 m |
| Volume | 1,68\*10^-5 m^3 |
| Density | 11340 kg/m^3 |
| Mass | 0.19kg |
| Air Coefficient | 0.47f |

The rope length is set to 0.1 meters but it can be changed in the program. Other values which could be set in the program are the radius and the initial angle of the balls.

Collision

If the distances between the two weights is less than the total radius of them a collision has been detected. The distances can be calculated using Pythagorean theorem(pic). Since a numerical method is used the weights can overlap each other and due can be stuck(pic). Therefore, when a collision is detected, the previous values ​​and positions of the weights is used. But to get the simulation more realistic the balls must be moved together so it looks like they collide. In our case the previous velocity vector is used as direction and with small steps size the weights transferred against each other until they intersect.(pic)

Next step is to find the normal plane between the weights, and this can be determined as the distance between center points of the weights. From the normal plane the collision plane can be calculated according to eq(something something)(pic).

Now the input velocity vectors can be describe in the normal- and the collision plane. To obtain this new vector an operation called vector dot product is used accordning to eq(somethingSomething). When this vectors are found the after collision veloctiy vector can be determined accordning to eq(something something)

// Calculate the scaler velocities of each object after the collision.  
 float n\_vel1\_after = ((n\_vel1 \* (object1.Mass - object2.Mass)) + (2 \* object2.Mass \* n\_vel2)) / (object2.Mass + object1.Mass);  
 float n\_vel2\_after = ((n\_vel2 \* (object2.Mass - object1.Mass)) + (2 \* object1.Mass \* n\_vel1)) / (object2.Mass + object1.Mass);  
 //float velObject2Tangent\_After = c\_vel2;  
 //float velObject1Tangent\_After = c\_vel1;  
  
 // Convert the scalers to vectors by multiplying by the normalised plane vectors.  
 Vector2 vec\_n\_vel2\_after = n\_vel2\_after \* normalPlane;  
 Vector2 vec\_c\_vel2 = c\_vel2 \* collisionPlane;  
 Vector2 vec\_n\_vel1\_after = n\_vel1\_after \* normalPlane;  
 Vector2 vec\_c\_vel1 = c\_vel1 \* collisionPlane;  
  
 // Combine the vectors back into a single vector in world space.  
 Vector2 vel1\_after = vec\_n\_vel1\_after + vec\_c\_vel1;  
 Vector2 vel2\_after = vec\_n\_vel2\_after + vec\_c\_vel2;

\begin{center}

\begin{table}

\begin{tabular}{| l | l |}

\hline

Radius & 0.02(m) \\ \hline

Volume & 1.68\*10^{-5}(m^3) \\ \hline

Density & 11340 (kg/m^3) \\ \hline

Mass & 0.19(kg)\\ \hline

Air Coefficient & 0.47 \\ \hline

\end{tabular}

\end{table}

\end{center}