

INVESTIGATION OF AN ELASTIC COLLISION SIMULATION WITH A BALL IN 1D AND 2D

S. ALI

Programmer, Data Analyst

G. PARASCHIV

Programmer, Data Analyst

A. XU

Programmer, Data Analyst

(Received 1 October 2020)

1.0 Theory

A collision between two objects can either be an elastic or inelastic collision depending on the physical results. In both cases, the laws of physics dictate that the linear and angular momentum of the system is conserved. However, in the case of elastic collisions, the kinetic energy of the system is conserved as well. This is because collisions within a system of two or more particles only involve internal forces. As such, the overall motion of the system, or the motion of the center of mass, is not affected given that the system is closed and isolated. While the conservation of momentum and conservation of kinetic energy are useful in calculating the initial and final conditions of a collision, they do not wholly represent the physical processes involved.

At a microscopic level, elastic collisions occur when the two objects compress at the point of impact. Chemical bonds in the two objects will stretch and result in a rebound effect. This action can be thought of as two springs acting in series. The springs will briefly decelerate the objects while the compression is occurring, and then later accelerate them as they decompress. This process happens over a very small period time (fractions of a second), and while the compression due to the impact is very slight, it is never zero. As it is often impractical to measure such microscopic properties in a lab and to then effect the calculations for the final conditions by hand, a simulation is required to model the motion of a collision.

This experiment sought to investigate and model an elastic collision between two particles using two stress balls. The stress balls are assumed to be of uniform density. The system of the two balls is assumed to be closed and isolated, meaning that any external forces, such as drag, are ignored for the purposes of effecting the laws of conservation. The spring constant of the two stress balls was given to be 426.

Given that a system is closed and isolated, the linear momentum of the system is conserved during a collision. In 2D collisions, the following equation is effected separately for the x and y components,

$$\vec{p}_{A_i} + \vec{p}_{B_i} = \vec{p}_{A_f} + \vec{p}_{B_f} \quad [1]$$

(Halliday et al, 2010)

Where:

\vec{p}_{A_i} = initial momentum of object A (kg m s⁻¹)

\vec{p}_{B_i} = initial momentum of object B (kg m s⁻¹)

\vec{p}_{A_f} = final momentum of object A (kg m s⁻¹)

\vec{p}_{B_f} = final momentum of object B (kg m s⁻¹)

Additionally, the kinetic energy of the system is conserved during an elastic collision,

$$T_{A_i} + T_{B_i} = T_{A_f} + T_{B_f} \quad [2]$$

(Halliday et al, 2010)

Where:

T_{A_i} = initial kinetic energy of object A (J)

T_{B_i} = initial kinetic energy of object B (J)

T_{A_f} = final kinetic energy of object A (J)

T_{B_f} = final kinetic energy of object B (J)

Using [3] and [4], the following equations can be derived for an elastic collision in one dimension,

$$\vec{v}_{A_f} = \frac{m_A - m_B}{m_A + m_B} \vec{v}_{A_o} + \frac{2m_B}{m_A + m_B} \vec{v}_{B_o} \quad [3]$$

$$\vec{v}_{B_f} = \frac{2m_A}{m_A + m_B} \vec{v}_{A_o} + \frac{m_B - m_A}{m_A + m_B} \vec{v}_{B_o} \quad [4]$$

(Halliday et al, 2010)

Where:

\vec{v}_{A_o} = initial velocity of object A (ms^{-1})

\vec{v}_{B_o} = initial velocity of object B (ms^{-1})

\vec{v}_{A_f} = final velocity of object A (ms^{-1})

\vec{v}_{B_f} = final velocity of object B (ms^{-1})

m_A = mass of object A (m)

m_B = mass of object B(m)

These equations are used when solving word problems as shown in the worked solutions below. To effect calculations in the simulation, an understanding of how springs apply forces is need. Hooke's law approximates the force that a spring applies on an object,

$$\vec{F}_s = -k\vec{x} \quad [5]$$

(Halliday et al, 2010)

Where:

\vec{F}_s = force applied by spring (N)

k = spring constant (kg s^{-3})

\vec{x} = displacement of spring/compression (m)

In this lab, there are two springs acting in series. The equation for the effective spring constant (the constant that is applied in Hooke's law) is given by the following equation,

$$k_{\text{eff}} = \frac{k_1 k_2}{k_1 + k_2} \quad [6]$$

(Weisstein, 2007)

Where:

k_{eff} = effective spring constant (kgs^{-2})

k_1 = spring constant of spring 1(kgs^{-2})

k_2 = spring constant of spring 2(kgs^{-2})

Newton's Second Law of Motion is used to calculate the acceleration of the balls due to the spring force,

$$\vec{a} = \frac{\vec{F}_{net}}{m} \quad [7]$$

(Halliday et al, 2010)

Where:

\vec{a} = linear acceleration (m s^{-2})

\vec{F}_{net} = net force (N)

m = mass of ball (kg)

Subsequently, the use of basic kinematic equations allows for the computation of velocity and displacement (van Bemmél, 2005).

The first kinematics equation is,

$$\vec{v} = \vec{v}_0 + \vec{a}t \quad [8]$$

(van Bemmél, 2020)

Where:

\vec{v} = Final velocity (ms^{-1})

\vec{v}_0 = Initial velocity (ms^{-1})

t = Time (s)

Due to the extremely small value of t^2 when substituted with dt , the term $\frac{1}{2}\vec{a}t^2$ is negligible and was excluded. The second kinematics equation is,

$$s = s_0 + \vec{v}_0t \quad [9]$$

(van Bemmél, 2020)

Where:

s = Final position (m)

s_0 = Initial position (m)

When the balls collide, torque is applied to each ball based on the tangential forces involved. In this application, the only force that is acting tangentially to the ball is the frictional force. This leads to the following equation for torque,

$$\vec{\tau} = \vec{r} \times (\hat{F}_f \mu_k F_s) \quad [10]$$

(Halliday et al, 2010)

Where:

$\vec{\tau}$ = torque (Nm)

\vec{r} = radius of ball (m)

\hat{F}_f = unit vector of frictional force

μ_k = coefficient of kinetic friction

F_s = magnitude of spring force (N)

The rotational inertia used in this lab is only taken into consideration when the balls are compressed during the collision. The balls in this state are approximated as ellipsoids. The rotational inertia of an ellipsoid is given by the equation,

$$I = \frac{1}{5}m(a^2 + b^2) \quad [11]$$

(Halliday et al, 2010)

Where:

I = rotational inertia (kg m^2)
 a = semi-major axis (m)
 b = semi-minor axis (m)

The rotational aspect of this lab requires slightly different equations. Rotational acceleration is defined by,

$$\vec{\alpha} = \frac{\vec{\tau}}{I} \quad [12]$$

(Halliday et al, 2010)

Where:

$\vec{\alpha}$ = rotational acceleration (rad s^{-2})

Angular velocity can be calculated in a similar way to translational velocity,

$$\vec{\omega}_f = \vec{\omega}_o + \vec{\alpha}t \quad [13]$$

(Halliday et al, 2010)

Where:

$\vec{\omega}_f$ = final rotational velocity (rad s^{-1})
 $\vec{\omega}_i$ = initial rotational velocity (rad s^{-1})

Contrary to the translational case, if friction is taken into consideration to allow for torque and rotational acceleration, rotational kinetic energy is no longer conserved. This is due to loss of energy as heat due to friction between the two balls. However, angular momentum is still conserved,

$$\vec{L}_{A_i} + \vec{L}_{B_i} = \vec{L}_{A_f} + \vec{L}_{B_f} \quad [14]$$

(Halliday et al, 2010)

Where:

\vec{L}_{A_i} = initial angular momentum of object A ($\text{kg m}^2 \text{s}^{-1}$)
 \vec{L}_{B_i} = initial angular momentum of object B ($\text{kg m}^2 \text{s}^{-1}$)
 \vec{L}_{A_f} = final angular momentum of object A ($\text{kg m}^2 \text{s}^{-1}$)
 \vec{L}_{B_f} = final angular momentum of object B ($\text{kg m}^2 \text{s}^{-1}$)

2.0 Method

Simulations involve repeating very simple calculations over a small interval dt . The interval in this lab was $1 \times 10^{-5} \text{s}$ as it was able to produce residuals of less than 0.03% which was deemed to be sufficiently accurate for the purposes of this lab. A dynamic time interval was considered, wherein dt would decrease during the collision. This was rejected as it would not significantly decrease the error beyond a reasonably small error.

The simulation was run over an arbitrarily chosen time interval such that there was excess left after the collision. After retrieving the data, the data was cut off such that the collision occurred at about halfway through the time frame.

The simulation functions by considering the net forces acting on each ball in an interval Δt , calculating the acceleration using [7], and then computing the velocity and position using kinematic equations. It repeats this process for every interval Δt . A force is only applied during the collision. The simulation detects collisions and determines the spring force using [5].

3.0 Program Listing

```

10/1/2020                                     Collision Simulator.cpp
1  /*
2  AP Physics Lab 1A
3  Collision Simulation
4  Sarah Ali, George Paraschiv, Arthur Xu
5  SPH4U0
6  */
7
8  // Define standard mass, radius, time increments
9  #define M 0.1
10 #define R 0.05
11 #define DT 0.0001
12
13 // Libraries
14 #include <iostream>
15 #include <math.h>
16 #include <string>
17 #include <fstream>
18
19 // Using standard namespace
20 using namespace std;
21
22 // Vector Class
23 class Vector
24 {
25 public:
26     double x, y, m, theta; // Variables for components, magnitude, and angle
27
28     Vector() // Default constructor for 0 vector
29     {
30         x = 0;
31         y = 0;
32         m = 0;
33         theta = 0;
34     }
35
36     Vector(double a, double b, bool cart) // Vector constructor with user-input
37     components
38     {
39         if (cart){
40             x = a;
41             y = b;
42             m = sqrt(a * a + b * b); // Calculates magnitude
43             theta = atan2(y, x);    // Calculates angle in radians
44         }
45         else{
46             m = a;
47             theta = b;
48             x = a*cos(b); // Calculates x component
49             y = a*sin(b); // Calculates y component
50         }
51     }
52
53     // Method that checks if the two vectors are parallel
54     bool isParallel(Vector v2){
55         // Since we are doing a simulation, the y values are often not true zeroes
56         // but instead are something like 10e-20
57         // Even if the values are not true zeroes, we still consider the vectors as
58         // parallel
59         // The less than 1e-10 is used to make sure that the value should be a true
60         // zero

```

```

10/1/2020 Collision Simulator.cpp
57     if(abs(y/x-v2.y/v2.x) < pow(1,-10)){
58         return true;
59     }
60     else{
61         return false;
62     }
63 }
64 };
65
66 // Overrides output operator to print vectors in readable format (x,y)
67 ostream &operator<<(ostream &strm, const Vector &v)
68 {
69     return strm << "(" << to_string(v.x) << ", " << to_string(v.y) << ")";
70 }
71
72 // Ball class
73 class Ball
74 {
75 public:
76     Vector s, v, a; // Linear kinematics vectors
77     double m, r, w, I, alpha, torque; // Mass and rotational variables: angular
78     // velocity, rotational inertia, angular acceleration, torque
79     double k; // Spring constant
80
81     Ball(Vector s0 = Vector(), Vector v0 = Vector(), double m0 = M, double r0 = R,
82         double k0 = 0, double w0 = 0) //constructor for ball that assigns user input or
83         // default values
84     {
85         s = s0;
86         v = v0;
87         m = m0;
88         r = r0;
89         w = w0;
90         k = k0;
91     }
92
93     void applyForce(Vector F, Vector compress, double torque) // Applies force and
94     // updates motion variables (part of flowchart)
95     {
96         a = Vector(F.x / m, F.y / m, true); // Calculate acceleration using Newtons
97         // 2nd Law: a = F/m
98
99         // Calculate rotational inertia for an ellipsoid: I = 0.2M(a^2 + b^2) where
100         // a and b are semi-minor axis
101         I = 0.2 * m * (pow(R - compress.m, 2) + pow(R, 2));
102         // Calculate angular acceleration using alpha = torque / I
103         alpha = torque / I;
104         // Updates linear velocity with kinematics equation v = v0 + a * t
105         v = Vector(v.x + a.x * DT, v.y + a.y * DT, true);
106         // Updates position with kinematics equation s = s0 + v0 * t -> for small
107         // values of t
108         s = Vector(s.x + v.x * DT, s.y + v.y * DT, true);
109         // Updates angular velocity with rotational kinematics equation w = w0 +
110         // alpha * t
111         w += alpha * DT;
112     }
113 };
114
115 // Driver function
116 int main()

```

```

10/1/2020 Collision Simulator.cpp
109 {
110     double sx, sy, vx, vy, w, m, r, T, c, k; // Temporary input values
111     double torque; // Value for torque
112     Vector spring = Vector(); // Spring force vector
113     Vector friction = Vector(); // Friction force vector
114     Vector forcenet = Vector(); // Net force vector
115
116     // Obtain User Input for intital conditions (part of the flowchart)
117     cout << "Enter the x component of ball 1's position: ";
118     cin >> sx;
119     cout << "Enter the y component of ball 1's position: ";
120     cin >> sy;
121     cout << "Enter the x component of ball 1's velocity: ";
122     cin >> vx;
123     cout << "Enter the y component of ball 1's velocity: ";
124     cin >> vy;
125     cout << "Enter the angular velocity of ball 1: ";
126     cin >> w;
127     cout << "Enter the spring constant of ball 1: ";
128     cin >> k;
129     cout << "Enter the mass of ball 1 (0 for default 0.1 kg): ";
130     cin >> m;
131     if (m == 0) // Allows user to choose default mass
132         m = M;
133
134     // Defines first ball using user input
135     Ball b1 = Ball(Vector(sx, sy, true), Vector(vx, vy, true), m, r, k, w);
136
137     cout << "Enter the x component of ball 2's position: ";
138     cin >> sx;
139     cout << "Enter the y component of ball 2's position: ";
140     cin >> sy;
141     cout << "Enter the x component of ball 2's velocity: ";
142     cin >> vx;
143     cout << "Enter the y component of ball 2's velocity: ";
144     cin >> vy;
145     cout << "Enter the angular velocity of ball 2: ";
146     cin >> w;
147     cout << "Enter the spring constant of ball 2: ";
148     cin >> k;
149     cout << "Enter the mass of ball 1 (0 for default 0.1 kg): ";
150     cin >> m;
151     if (m == 0) // Allows user to choose default mass
152         m = M;
153
154     // Defines second ball using user input
155     Ball b2 = Ball(Vector(sx, sy, true), Vector(vx, vy, true), m, r, k, w);
156
157     // Coefficient of friction for the collision
158     cout << "Enter the coefficient of friction between the two balls: ";
159     cin >> c;
160
161     // Time limit of program in seconds
162     cout << "Enter the time limit: ";
163     cin >> T;
164
165     // Creates a csv output file
166     ofstream f("Output.csv");
167
168     // Header for output file

```

```

10/1/2020 Collision Simulator.cpp
169 f << "Time" << "," << "Ball 1 Position" << "," << "Ball 2 Position" << "," <<
    "Ball 1 Velocity" << "," << "Ball 2 Velocity" << "," << "Ball 1 Angular Velocity" <<
    "," << "Ball 1 Angular Acceleration" << "," << "Ball 2 Angular Velocity" << "," <<
    "Ball 2 Angular Acceleration" << "\n";
170
171 // Loop for simulation during time interval with dt increments
172 // Have we iterated through the indicated number of steps (part of flowchart)
173 for (double t = 0; t < T; t += DT)
174 {
175     /* Calculates the relative position vector from ball 1 to ball 2.
176     Calculates compression vector using radii and relative position vector (part
    of flowchart) */
177     Vector srel = Vector(b1.s.x - b2.s.x, b1.s.y - b2.s.y, true);
178     Vector compression ((b1.r+b2.r - srel.m), srel.theta, false);
179
180     double keff = ((b1.k*b2.k)/(b1.k+b2.k));
181
182     if (compression.m > 0) // Checks if the balls are in contact (part of
    flowchart)
183     {
184         // Calculates spring force using  $F = kx$  (part of flowchart)
185         spring = Vector(compression.x * keff, compression.y * keff, true);
186         /* Since the force of friction is perpendicular to the spring force, then
    the spring force is just the normal force.
187         If they are perpendicular, their x and y coordinates are simply
    switched, with a negative x coordinate in the friction vector.
188         Note that we must also multiply by friction coefficient because Force of
    friction = (coefficient of friction)*(normal force). (part of flowchart)*/
189         friction = Vector(-spring.y*c, spring.x*c, true);
190
191         /* torque = (radius)x(net tangent force)
192            = (radius)x(force of friction)
193         The spring force is always radial and does not apply any torque. (part
    of flowchart) */
194         torque = (R-compression.m)*cos(compression.theta)*friction.y-(R-
    compression.m)*sin(compression.theta)*friction.x; // This is the cross product
    definition in 2D
195
196         /* If friction and the translational velocity are perpendicular, there
    is no frictional force that affects translational velocity, because no component of
    translational velocity is parallel to friction.
197         By definition, friction opposes velocity, but this is not possible if
    they are perpendicular.
198         Therefore friction is not taken into consideration into the net force.
199         Since the relative position is perpendicular to the friction, then if
    velocity is parallel to the relative position, the translational velocity will then
    be perpendicular to the friction force (part of flowchart)*/
200         if(srel.isParallel(b1.v) == true){
201             // If no friction force then by default net force is spring force
    (part of flowchart)
202             forcenet = spring;
203         }
204         else{ //If friction is present the following is true (part of flowchart)
205             // Net force adds the spring force to the friction force
206             forcenet = Vector(spring.x + friction.x, spring.y + friction.y,
    true);
207         }
208     }
209     else
210     {

```


10/1/2020

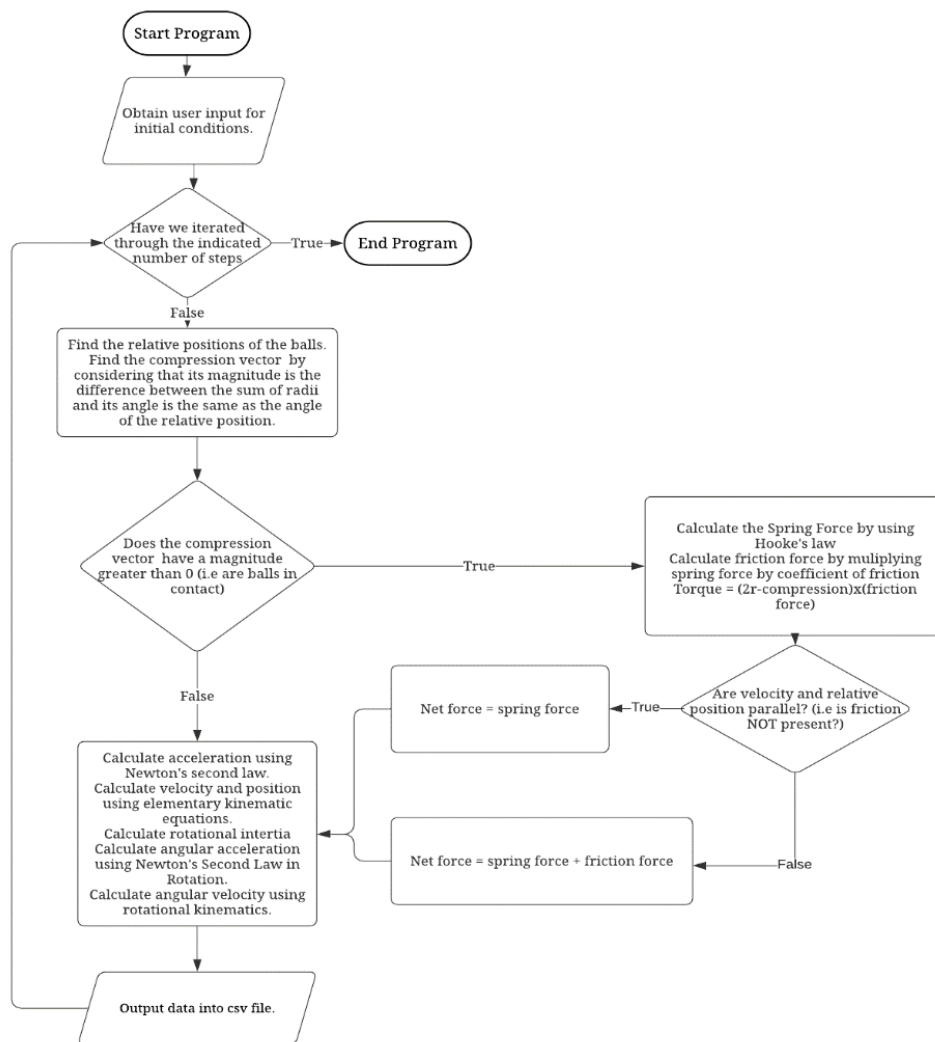
Collision Simulator.cpp

```

211         // Resets values to 0 if balls are not in contact
212         forcenet = Vector();
213         torque = 0;
214         compression = Vector();
215     }
216
217     // Applies force and updates motion variables (part of flowchart)
218     b1.applyForce(forcenet, compression, torque);
219     b2.applyForce(Vector(-forcenet.x, -forcenet.y, true), compression, -torque);
220
221     // Outputs values to csv file (part of flowchart)
222     f << to_string(t) << "," << b1.s << "," << b2.s << "," << b1.v << "," <<
b2.v << "," << b1.w << "," << b1.alpha << "," << b2.w << "," << b2.alpha << "\n";
223 }
224
225 f.close(); // Closes the output stream
226 }

```

3.1 Flowchart



3.0 Sample Problems

The following sections discuss problems concerning the four cases of a 1-D and a 2-D elastic collision with a resting target and a moving target. The simulation was run given the initial conditions from these sample problems.

4.1 1D Collision

The following are problems concerning a 1-D elastic collision with a resting and a moving target.

4.1.1 Resting Target

The 1-D collision simulation with a resting target was based on the following question from the Fundamentals of Physics 9th edition textbook by Halliday, D., Resnick, R., Walker, J. (2010).

•61 SSM A cart with mass 340 g moving on a frictionless linear air track at an initial speed of 1.2 m/s undergoes an elastic collision with an initially stationary cart of unknown mass. After the collision, the first cart continues in its original direction at 0.66 m/s. (a) What is the mass of the second cart? (b) What is its speed after impact? (c) What is the speed of the two-cart center of mass?

The worked solution is as follows:

61. Let m_1 be the mass of the cart that is originally moving, v_{1i} be its velocity before the collision, and v_{1f} be its velocity after the collision. Let m_2 be the mass of the cart that is originally at rest and v_{2f} be its velocity after the collision. Conservation of linear momentum gives $m_1 v_{1i} = m_1 v_{1f} + m_2 v_{2f}$. Similarly, the total kinetic energy is conserved and we have

$$\frac{1}{2} m_1 v_{1i}^2 = \frac{1}{2} m_1 v_{1f}^2 + \frac{1}{2} m_2 v_{2f}^2.$$

Solving for v_{1f} and v_{2f} , we obtain:

$$v_{1f} = \frac{m_1 - m_2}{m_1 + m_2} v_{1i}, \quad v_{2f} = \frac{2m_1}{m_1 + m_2} v_{1i}$$

The speed of the center of mass is $v_{\text{com}} = \frac{m_1 v_{1i} + m_2 v_{2i}}{m_1 + m_2}$.

(a) With $m_1 = 0.34$ kg, $v_{1i} = 1.2$ m/s and $v_{1f} = 0.66$ m/s, we obtain

$$m_2 = \frac{v_{1i} - v_{1f}}{v_{1i} + v_{1f}} m_1 = \left(\frac{1.2 \text{ m/s} - 0.66 \text{ m/s}}{1.2 \text{ m/s} + 0.66 \text{ m/s}} \right) (0.34 \text{ kg}) = 0.0987 \text{ kg} \approx 0.099 \text{ kg}.$$

(b) The velocity of the second cart is:

$$v_{2f} = \frac{2m_1}{m_1 + m_2} v_{1i} = \left(\frac{2(0.34 \text{ kg})}{0.34 \text{ kg} + 0.099 \text{ kg}} \right) (1.2 \text{ m/s}) = 1.9 \text{ m/s}.$$

The problem stated that the collision was elastic and that the target was initially stationary. Thus, it is appropriate for this section. The results of the simulation are given in Figures 1-3.

ID	t (s)	\vec{s}_A (m)	\vec{s}_B (m)	\vec{v}_A (ms ⁻¹)	\vec{v}_B (ms ⁻¹)
1	0	0	+1	+1.2	0
2	0.2	+0.24	+1	+1.2	0
3	0.4	+0.48	+1	+1.2	0
4	0.6	+0.72	+1	+1.2	0
5	0.8	+0.904079	+1.19264	+0.660042	+1.86004
6	1	+1.03609	+1.56464	+0.660042	+1.86004
7	1.2	+1.1681	+1.93665	+0.660042	+1.86004
8	1.4	+1.3001	+2.30866	+0.660042	+1.86004
9	1.6	+1.43211	+2.68067	+0.660042	+1.86004
10	1.8	+1.56412	+3.05267	+0.660042	+1.86004

Fig 1. Position and Velocity for 1D Collision with Stationary Target. The velocity of ball A halves after the collision, while ball B gains a greater velocity.

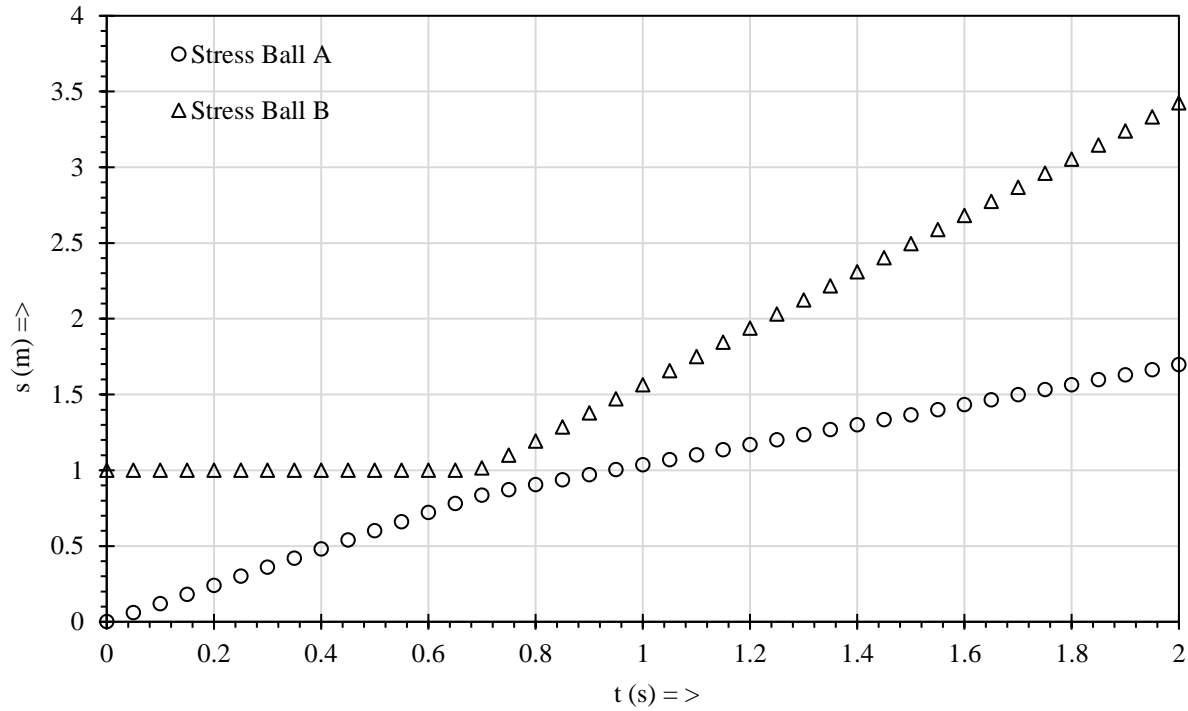


Fig 2. Position vs Time for 1D Collision with Stationary Target. The balls collide at $t = 0.6668$ s. The collision caused ball A to slow down and caused ball B to develop a positive velocity, as shown by the slopes.

ID	Solution Type	\vec{v}_{A_o} (ms ⁻¹)	\vec{v}_{B_o} (ms ⁻¹)	\vec{v}_{A_f} (ms ⁻¹)	\vec{v}_{B_f} (ms ⁻¹)
1	Simulation	+1.2	0	0.66004	1.86004
2	Solution	+1.2	0	0.66	1.86
3	Residuals	0%	0%	0.0067%	0.002%

Fig 3. Comparison of Simulation Results and Problem Solutions for 1D Collision with Stationary Target.
The residuals are very small, verifying the accuracy of the code.

4.1.2 Moving Target

The 1-D collision simulation with a moving target was based on the following question from the Fundamentals of Physics 9th edition textbook by Halliday, D., Resnick, R., Walker, J. (2010).

•62 Two titanium spheres approach each other head-on with the same speed and collide elastically. After the collision, one of the spheres, whose mass is 300 g, remains at rest. (a) What is the mass of the other sphere? (b) What is the speed of the two-sphere center of mass if the initial speed of each sphere is 2.00 m/s?

The worked solution is as follows:

62. (a) Let m_1 be the mass of one sphere, v_{1i} be its velocity before the collision, and v_{1f} be its velocity after the collision. Let m_2 be the mass of the other sphere, v_{2i} be its velocity before the collision, and v_{2f} be its velocity after the collision. Then, according to Eq. 9-75,

$$v_{1f} = \frac{m_1 - m_2}{m_1 + m_2} v_{1i} + \frac{2m_2}{m_1 + m_2} v_{2i}.$$

Suppose sphere 1 is originally traveling in the positive direction and is at rest after the collision. Sphere 2 is originally traveling in the negative direction. Replace v_{1i} with v , v_{2i} with $-v$, and v_{1f} with zero to obtain $0 = m_1 - 3m_2$. Thus,

$$m_2 = m_1 / 3 = (300 \text{ g}) / 3 = 100 \text{ g}.$$

(b) We use the velocities before the collision to compute the velocity of the center of mass:

$$v_{\text{com}} = \frac{m_1 v_{1i} + m_2 v_{2i}}{m_1 + m_2} = \frac{(300 \text{ g})(2.00 \text{ m/s}) + (100 \text{ g})(-2.00 \text{ m/s})}{300 \text{ g} + 100 \text{ g}} = 1.00 \text{ m/s}.$$

The problem stated that the collision was elastic and that both stress balls were initially in motion. Thus, it is appropriate for this section. The final velocity of ball B was calculated using [4] to be $+4 \text{ ms}^{-1}$. The results of the simulation are given in Figures 4-6.

ID	t (s)	\vec{s}_A (m)	\vec{s}_B (m)	\vec{v}_A (ms^{-1})	\vec{v}_B (ms^{-1})
1	0	0	+2	+2	-2
2	0.2	+0.2	+1.8	+2	-2
3	0.4	+0.4	+1.6	+2	-2
4	0.6	+0.6	+1.4	+2	-2
5	0.8	+0.8	+1.2	-1.77E-6	+4.00001
6	1	+0.929475	+1.21157	-1.77E-6	+4.00001
7	1.2	+0.929475	+1.61157	-1.77E-6	+4.00001
8	1.4	+0.929475	+2.01158	-1.77E-6	+4.00001
9	1.6	+0.929475	+2.41158	-1.77E-6	+4.00001
10	1.8	+0.929475	+2.81158	-1.77E-6	+4.00001

Fig 4. Position and Velocity for 1D Collision with Moving Target. The simulation result for the final velocity of ball A is a de facto zero, thus the position of ball A after the collision remains constant.

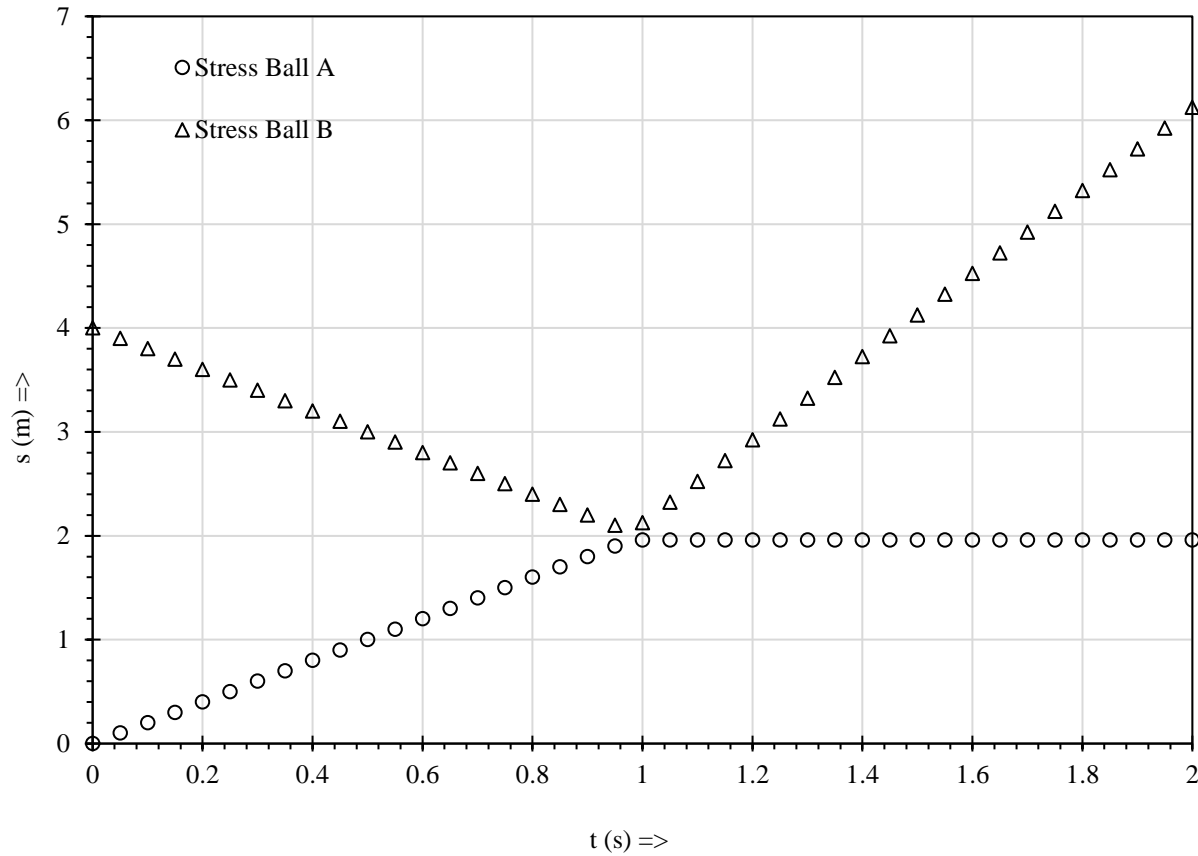


Fig 5. Position vs Time for 1D Collision with Moving Target. The balls collide at $t = 0.9002\text{s}$. The collision caused ball A to slow down and reach a final velocity of 0.

ID	Solution Type	\vec{v}_{A_o} (ms ⁻¹)	\vec{v}_{B_o} (ms ⁻¹)	\vec{v}_{A_f} (ms ⁻¹)	\vec{v}_{B_f} (ms ⁻¹)
1	Simulation	+2	-2	-1.77E-6	4.00001
2	Solution	+2	-2	0	4
3	Residuals	0%	0%	0%	0.00025%

Fig 6. Comparison of Simulation Results and Problem Solutions for 1D Collision with Moving Target.
 Since the final velocity of ball A is a de facto zero, the residual from the solution is also a de facto zero.

4.2 2D Collision

The following are problems concerning a 2-D elastic collision with a resting and a moving target.

4.2.1 Glancing Resting Target

The 2-D collision simulation with a resting target was based on the following question from the Fundamentals of Physics 9th edition textbook by Halliday, D., Resnick, R., Walker, J. (2010).

95 SSM In the arrangement of Fig. 9-21, billiard ball 1 moving at a speed of 2.2 m/s undergoes a glancing collision with identical billiard ball 2 that is at rest. After the collision, ball 2 moves at speed 1.1 m/s, at an angle of $\theta_2 = 60^\circ$. What are (a) the magnitude and (b) the direction of the velocity of ball 1 after the collision? (c) Do the given data suggest the collision is elastic or inelastic?

The worked solution is as follows:

95. The mass of each ball is m , and the initial speed of one of the balls is $v_{1i} = 2.2$ m/s. We apply the conservation of linear momentum to the x and y axes, respectively:

$$mv_{1i} = mv_{1f} \cos \theta_1 + mv_{2f} \cos \theta_2$$

$$0 = mv_{1f} \sin \theta_1 - mv_{2f} \sin \theta_2.$$

The mass m cancels out of these equations, and we are left with two unknowns and two equations, which is sufficient to solve.

(a) Solving the simultaneous equations leads to

$$v_{1f} = \frac{\sin \theta_2}{\sin(\theta_1 + \theta_2)} v_{li}, \quad v_{2f} = \frac{\sin \theta_1}{\sin(\theta_1 + \theta_2)} v_{li}.$$

Since $v_{2f} = v_{li} / 2 = 1.1 \text{ m/s}$ and $\theta_2 = 60^\circ$, we have

$$\frac{\sin \theta_1}{\sin(\theta_1 + 60^\circ)} = \frac{1}{2} \Rightarrow \tan \theta_1 = \frac{1}{\sqrt{3}}$$

or $\theta_1 = 30^\circ$. Thus, the speed of ball 1 after collision is

$$v_{1f} = \frac{\sin \theta_2}{\sin(\theta_1 + \theta_2)} v_{li} = \frac{\sin 60^\circ}{\sin(30^\circ + 60^\circ)} v_{li} = \frac{\sqrt{3}}{2} v_{li} = \frac{\sqrt{3}}{2} (2.2 \text{ m/s}) = 1.9 \text{ m/s}.$$

(b) From the above, we have $\theta_1 = 30^\circ$, measured *clockwise* from the +x-axis, or equivalently, -30° , measured *counterclockwise* from the +x-axis.

(c) The kinetic energy before collision is $K_i = \frac{1}{2} m v_{li}^2$. After the collision, we have

$$K_f = \frac{1}{2} m (v_{1f}^2 + v_{2f}^2).$$

Substituting the expressions for v_{1f} and v_{2f} found above gives

$$K_f = \frac{1}{2} m \left[\frac{\sin^2 \theta_2}{\sin^2(\theta_1 + \theta_2)} + \frac{\sin^2 \theta_1}{\sin^2(\theta_1 + \theta_2)} \right] v_{li}^2.$$

Since $\theta_1 = 30^\circ$ and $\theta_2 = 60^\circ$, $\sin(\theta_1 + \theta_2) = 1$ and $\sin^2 \theta_1 + \sin^2 \theta_2 = \sin^2 \theta_1 + \cos^2 \theta_1 = 1$, and indeed, we have $K_f = \frac{1}{2} m v_{li}^2 = K_i$, which means that energy is conserved.

The problem stated that the collision was elastic and that one stress balls was initially in motion and the other at rest. Thus, it is appropriate for this section. The initial positions of the 2 balls are not given in the problem. The target (ball B) was assumed to be at the origin. The horizontal component of ball A's initial position simply had to be to the left of ball B since it has an initial horizontal velocity to the right. The initial y position of ball B had to be found using trial and error through the simulation until the final velocities matched those provided by the worked solution. The results of the simulation are given in Figures 7-9.

ID	t (s)	\vec{s}_A (m,m)	\vec{s}_B (m,m)	\vec{v}_A (ms ⁻¹ , ms ⁻¹)	\vec{v}_B (ms ⁻¹ , ms ⁻¹)
1	0	(-2,-0.17325)	(0,0)	(2.2,0)	(0,0)
2	0.2	(-1.56,-0.17325)	(0,0)	(2.2,0)	(0,0)
3	0.4	(-1.12,-0.17325)	(0,0)	(2.2,0)	(0,0)
4	0.6	(-0.68,-0.17325)	(0,0)	(2.2,0)	(0,0)
5	0.8	(-0.24,-0.17325)	(0,0)	(2.2,0)	(0,0)
6	1.0	(0.12506,-0.303)	(0.074944,0.12988)	(1.64503,-0.9503)	(0.54989, 0.9526)
7	1.2	(0.455119,-0.494)	(0.18488,0.3204)	(1.64503,-0.9503)	(0.54989, 0.9526)
8	1.4	(0.785182,-0.684)	(0.29482,0.51091)	(1.64503,-0.9503)	(0.54989, 0.9526)
9	1.6	(1.1152,-0.8746)	(0.40476,0.7014)	(1.64503,-0.9503)	(0.54989, 0.9526)
10	1.8	(1.4453,-1.06514)	(0.51469,0.8919)	(1.64503,-0.9503)	(0.54989, 0.9526)

Fig 7. Position and Velocity for 2-D Collision with Resting Target Graph. The change in the x velocities of the two balls is little relative to the change in the y velocities.

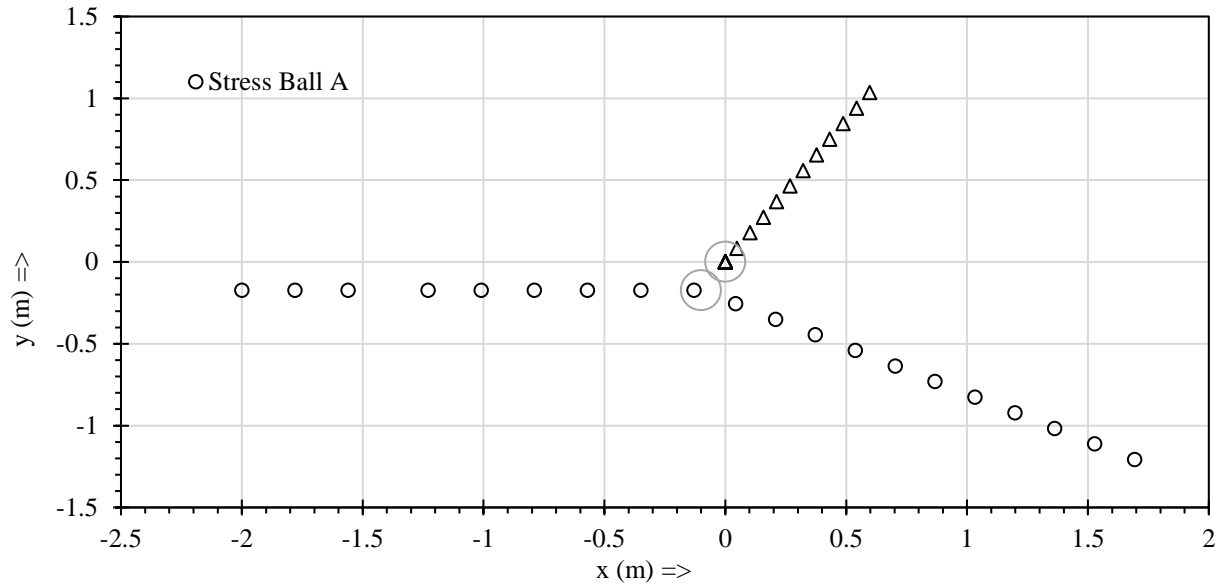


Fig 8. Y vs X for 2-D Collision with Resting Target. The balls collide at the origin at $t = 0.8636s$. Since ball A had a slightly smaller y position than ball B, ball A developed a negative y velocity while B developed a positive y velocity. The light grey circles indicate the boundaries of the balls.

ID	Solution Type	\vec{v}_{A_o} (ms ⁻¹ , ms ⁻¹)	\vec{v}_{B_o} (ms ⁻¹ , ms ⁻¹)	\vec{v}_{A_f} (ms ⁻¹ , ms ⁻¹)	\vec{v}_{B_f} (ms ⁻¹ , ms ⁻¹)
1	Simulation	(2.2,0)	(0,0)	(1.64503,-0.95028)	(0.54989, 0.95258)
2	Solution	(2.2,0)	(0,0)	(1.64545,-0.95)	(0.55,0.95262)
3	Residuals	0	0	(0.03,0.03)%	(0.02,0.004)%

Fig 9. Comparison of Simulation Results and Problem Solutions for 2D Collision with Resting Target. The residuals are slightly larger than the residuals for the 1D results, suggesting lower accuracy for more complex simulations such as glancing collisions.

4.2.2 Moving Target

The 2-D collision simulation with a moving target was based on the following question from the Fundamentals of Physics 9th edition textbook by Halliday, D., Resnick, R., Walker, J. (2010).

84 Figure 9-73 shows an overhead view of two particles sliding at constant velocity over a frictionless surface. The particles have the same mass and the same initial speed $v = 4.00$ m/s, and they collide where their paths intersect. An x axis is arranged to bisect the angle between their incoming paths, such that $\theta = 40.0^\circ$. The region to the right of the collision is divided into four lettered sections by the x axis and four numbered dashed lines. In what region or along what line do the particles travel if the collision is (a) completely inelastic, (b) elastic, and (c) inelastic? What are their final speeds if the collision is (d) completely inelastic and (e) elastic?

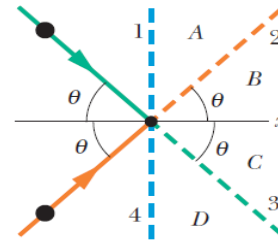


Figure 9-73 Problem 84.

The worked solution is as follows:

84. (a) This is a highly symmetric collision, and when we analyze the y -components of momentum we find their net value is zero. Thus, the stuck-together particles travel along the x axis.

(b) Since it is an elastic collision with identical particles, the final speeds are the same as the initial values. Conservation of momentum along each axis then assures that the angles of approach are the same as the angles of scattering. Therefore, one particle travels along line 2, the other along line 3.

(c) Here the final speeds are less than they were initially. The total x -component cannot be less, however, by momentum conservation, so the loss of speed shows up as a decrease in their y -velocity-components. This leads to smaller angles of scattering. Consequently, one particle travels through region B, the other through region C; the paths are symmetric about the x -axis. We note that this is intermediate between the final states described in parts (b) and (a).

(d) Conservation of momentum along the x -axis leads (because these are identical particles) to the simple observation that the x -component of each particle remains constant:

$$v_{fx} = v \cos \theta = 3.06 \text{ m/s.}$$

(e) As noted above, in this case the speeds are unchanged; both particles are moving at 4.00 m/s in the final state.

The problem stated that the collision was elastic and that the target was initially moving. Thus, it is appropriate for this section. The initial positions were not given by the problem but can

be mathematically determined. Since the balls have the same speed, β and collide at the origin, the initial x positions must be the same. With similar logic, it can be shown that the y positions are symmetrical across the x axis. The magnitude of the position vector of each ball was assumed to be 1. The results of the simulation are given in Figures 10-12.

ID	t (s)	\vec{s}_A (m,m)	\vec{s}_B (m,m)	\vec{v}_A (ms ⁻¹ , ms ⁻¹)	\vec{v}_B (ms ⁻¹ , ms ⁻¹)
1	0	(-0.7657, 0.6425)	(-0.7657,-0.6425)	(3.0642, -2.57115)	(3.0642, 2.57115)
2	0.06	(-0.58219, 0.4885)	(-0.58219,-0.4885)	(3.0642, -2.57115)	(3.0642, 2.57115)
3	0.12	(-0.3983, 0.33425)	(-0.3983, -0.3343)	(3.0642, -2.57115)	(3.0642, 2.57115)
4	0.18	(-0.2145, 0.17998)	(-0.2145,-0.17998)	(3.0642, -2.57115)	(3.0642, 2.57115)
5	0.24	(-0.0306, 0.06255)	(-0.0306,-0.0626)	(3.0642, 0.788657)	(3.0642,-0.788657)
6	0.3	(0.15321, 0.2048)	(0.1532,-0.205)	(3.0642, 2.57116)	(3.0642, -2.57116)
7	0.36	(0.3371, 0.3591)	(0.33706,-0.359)	(3.0642, 2.57116)	(3.0642, -2.57116)
8	0.42	(0.5209, 0.5133)	(0.5209,-0.513)	(3.0642, 2.57116)	(3.0642, -2.57116)
9	0.48	(0.70476, 0.6676)	(0.70476,-0.6676)	(3.0642, 2.57116)	(3.0642, -2.57116)
10	0.54	(0.8886, 0.8219)	(0.8886,-0.82188)	(3.0642, 2.57116)	(3.0642, -2.57116)

Fig 10. Position and Velocity for 2-D Collision with Moving Target. The final velocity of ball B is equal to the initial velocity of ball A, and vice versa, due to the balls having the same mass.

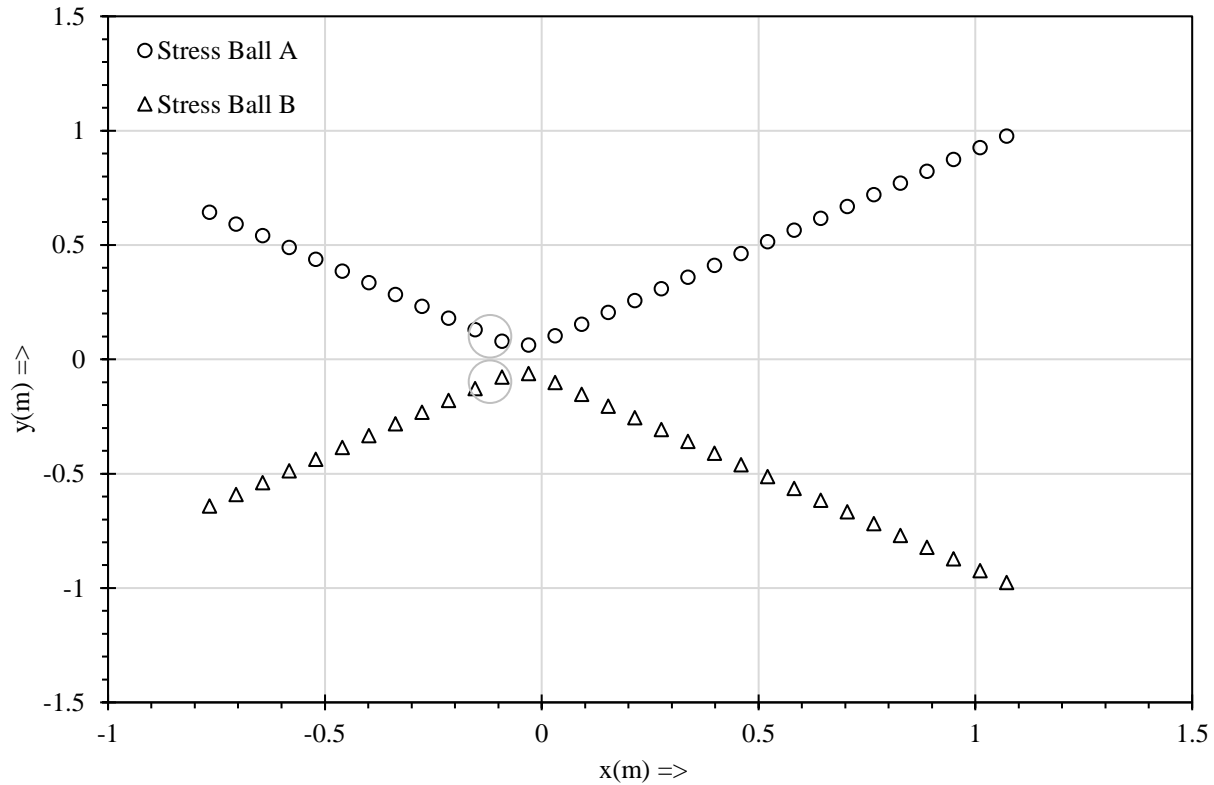


Fig 11. Y vs X for 2-D Collision with Moving Target. The balls collide at the origin at $t = 0.2146s$. Since the 2 balls have the same mass, they only switched velocities after the collision. The light grey circles indicate the boundaries of the balls.

ID	Solution Type	\vec{v}_{A_o} (ms ⁻¹ , ms ⁻¹)	\vec{v}_{B_o} (ms ⁻¹ , ms ⁻¹)	\vec{v}_{A_f} (ms ⁻¹ , ms ⁻¹)	\vec{v}_{B_f} (ms ⁻¹ , ms ⁻¹)
1	Simulation	(3.0642,-2.57115)	(3.064, 2.57115)	(3.064, 2.5712)	(3.064,-2.5712)
2	Solution	(3.0642,-2.57115)	(3.064, 2.57115)	(3.064, 2.5712)	(3.064,-2.5712)
3	Residuals	0	0	(0, 0.0004)%	(0,0.0004)%

Fig 12. Comparison of Simulation Results and Problem Solutions for 2D Collision with Moving Target.
The residuals are noticeably smaller than the residuals in the previous 2D problem, suggesting higher accuracy.

5.0 Rotation

To simulate the rotational motion of the two balls, the force of friction needs to be considered as a tangential force is required to produce a torque, which will cause angular acceleration. Friction is the only force that is tangent to the radius of the balls because the spring force is radial to the centre of each ball. The frictional force is perpendicular to the spring force and antiparallel to the rotational force at the point of contact. It will also act in the opposite direction of the component of the velocity that is parallel to the friction, meaning that it will affect the translational motion of the balls as well in 2D collisions.

To calculate the force of friction, the spring force was taken as the normal force. The coefficient of kinetic friction could be set to any arbitrary value in the range [0,1]. The coefficient of kinetic friction of a stress ball was approximated to be 0.05 for the simulation. The torque being applied to each ball was then taken as the 2D pseudo cross-product of the radius at the point of contact and the force of friction using [10]. The torque was then applied to each ball to produce angular acceleration using [12]. The rotational inertia was taken as a dynamic value due to the compression of the balls at different points in time and calculated using [11]. The angular velocity was then updated using the kinematic equation [13].

An interesting outcome of this condition is that head on collision the friction will not affect the translational motion. The linear velocity is parallel to the spring force and perpendicular to the friction, meaning that no component of linear velocity is parallel to friction. Thus, in the simulation, an if statement must be used to check if during the collision, the translational velocity is parallel to the spring force.

The sample problem in 4.2.1 was simulated using arbitrary initial angular velocities. The results are given in Figures 13-15.

ID	t (s)	ω_A (rad s ⁻¹)	L_A (kgm ² s ⁻¹)	ω_B (rad s ⁻¹)	L_B (kgm ² s ⁻¹)	L_{sys} (kgm ² s ⁻¹)	\vec{v}_A (ms ⁻¹ ,ms ⁻¹)	\vec{v}_B (ms ⁻¹ ,ms ⁻¹)
1	0	+1	+0.0004	-2	-0.0008	-0.0004	(2,0)	(0,0)
2	0.3	+1	+0.0004	-2	-0.0008	-0.0004	(2,0)	(0,0)
3	0.6	+1.7226	+0.000689	-2.7226	-0.00109	-0.0004	(0.279,-0.792)	(1.7206,0.7924)
4	0.9	+1.7226	+0.000689	-2.72255	-0.00109	-0.0004	(0.279,-0.792)	(1.7206,0.7924)

Fig 13. Angular Velocity and Momentum of Ball A and B. The angular momentum of the system is conserved.

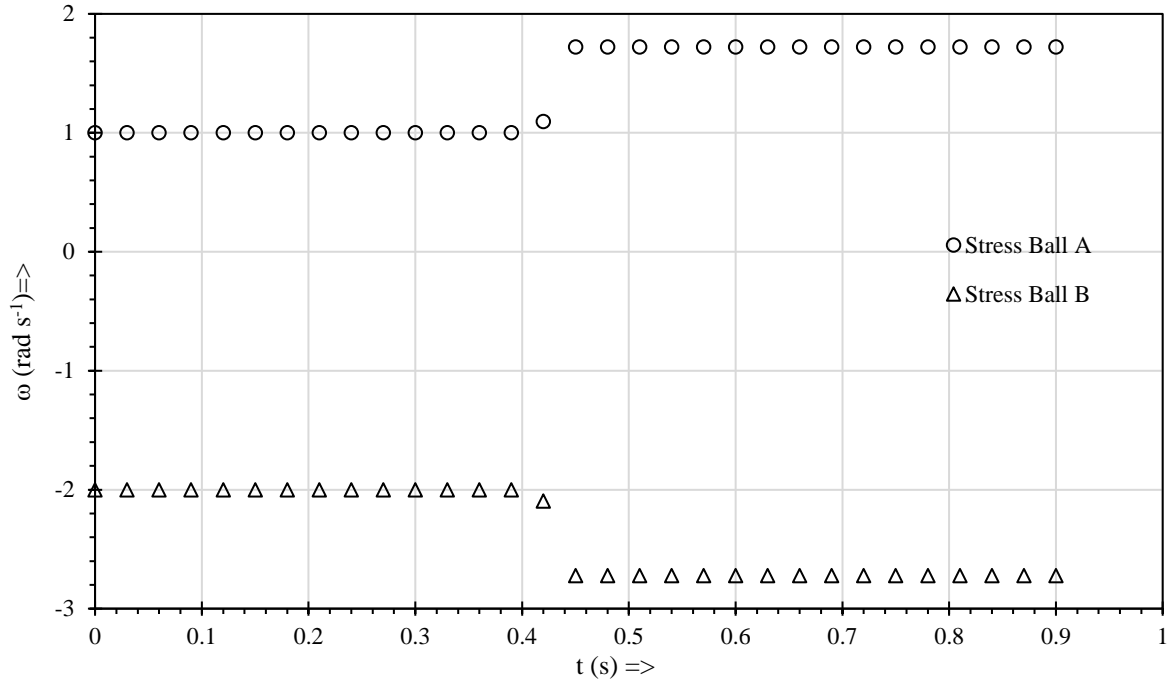


Fig 14. Angular Velocity vs Time in Glancing Collision. Ball A increased its angular velocity while ball B decreased its angular velocity by the same amount obeying the conservation of linear momentum.

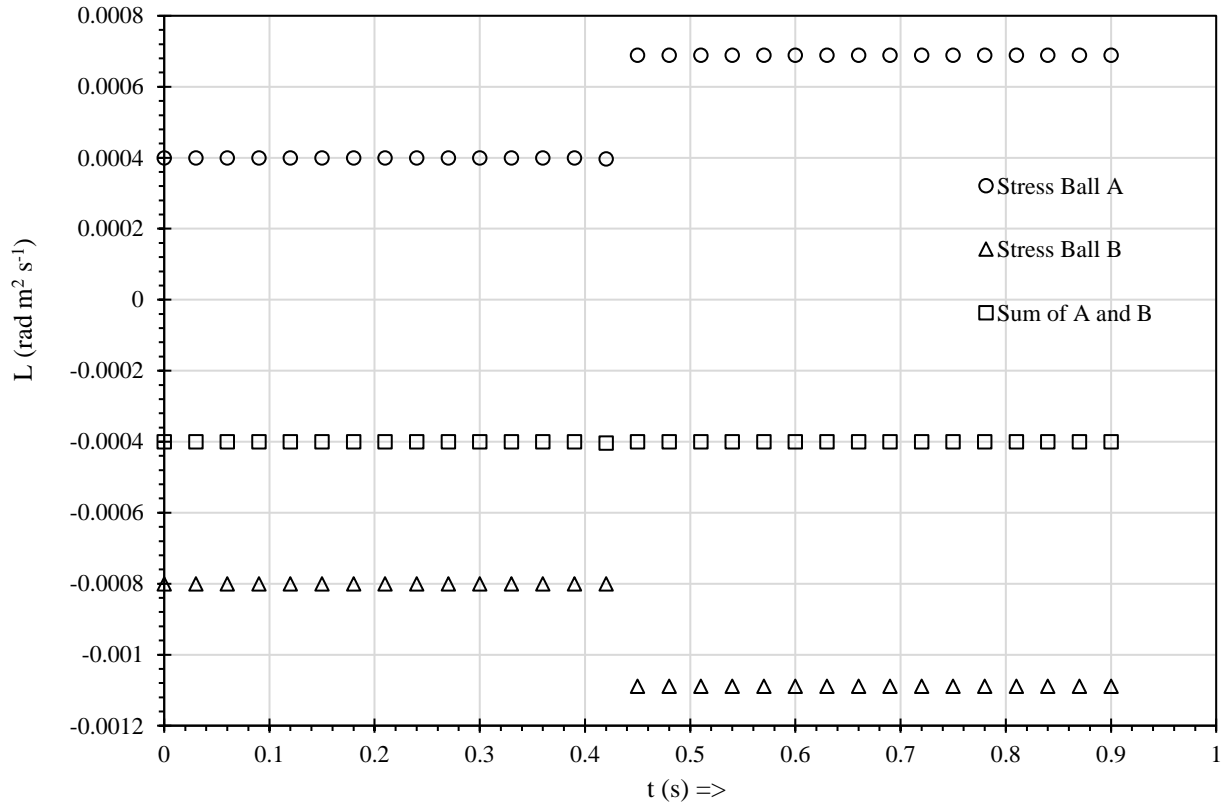


Fig 15. Angular Momentum vs Time in Glancing Collision This graph displays the angular momentum of the glancing collision displayed in Fig 13. A third plot is displayed to show that the total angular momentum is constant throughout.

6.0 Sources

van Bemmél, H., "Simulations", <http://www.hmvb.org/Simulation%20Primer.pdf>, 2005.

van Bemmél H., "Kinematics Notes", 2020.

Halliday, D., Resnick, R., Walker, J. (HRW), *Fundamentals of Physics*, John Wiley & Sons Inc., 2010.

Weisstein, E., <https://scienceworld.wolfram.com/physics/SpringsTwoSpringsinSeries.html>, 2007.

Weisstein, E., <https://scienceworld.wolfram.com/physics/MomentofInertiaEllipsoid.html>, 1995.

FORMAL REPORT MARKING RUBRIC – LABORATORY PAPERS APC2021

Lab: 1 2 3 4 5 **Submit signature page on reverse**


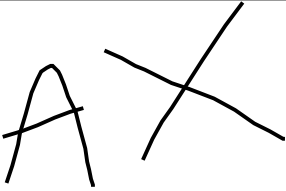
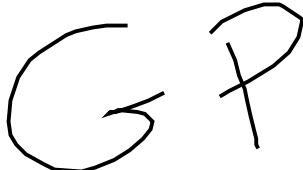
Student1 : Sarah Ali Student2 : George Paraschiv Student3 : Arthur Xu

Student4 : _____ Student5 : _____

SUB CATEGORY		DESCRIPTION			WT%	LETTER GRADE
SYNTAX		Spelling and Grammar including proper tense of expression. Was jargon used appropriately?			10	
FORM – GRAPHS AND CHARTS		Do your graphs and tables follow the guidelines? Did you put a caption beneath each one? Are they numbered? Are they well presented? Are they necessary? Your tables should not break at the end of a page. Did you remember the index column? Are your graphs scaled properly and with the proper format? Have you included error bars as required?			7	
FORM – EQUATIONS AND MARGINS		Are your equations properly presented? Are the variables defined with the expected units included? Are variables only defined on first use? Are your equations numbered with the number well off to the right? Did you source them as required? Are subscripts used correctly? Do your margin and columns conform to the rules? Is the font properly chosen? Are you using the correct size of paper and did you set up the first page as per the exemplar. Does your report conform to the length rules? Is your report free from silly computer glitches such as widows, orphans, and large gaps in the text? You must fix all of this. Did you email the .pdf copy of your work to your instructor prior to the due time? Did you print your work on BOTH sides of the paper? Did you submit a single page on the back of the marking rubric, detailing the work of the team members and include signatures of all?			7	
FORM – CITATIONS AND SOURCES		Have you cited any facts not fairly earned by your team? Are they cited in the proper manner described in the lab manual? Did you list all your sources? Are the sources conveyed in the proper format? Did you include at least TWO text sourcings?			7	
UNCERTAINTY		Have you indicated the uncertainty of your measured, computed and regressed values? Has this been done according to the rules as stipulated in the manual? Have your uncertainties been stated in the proper format?			10	
ABSTRACT		Is this abstract written in the proper form? Is it a fair description of this work and its accomplishments? Is it a reasonable length? Does the abstract use some of the most impressive numerical values to buttress its claims?			5	
METHODOLOGY		How have you used the equipment available to you? Have you maximized the precision and possibly the accuracy of your work? Did you know that you were doing this? Did you waste undue time on lengthy, but not effective or productive techniques?			17	
BASIC CONTENT		Were the basic questions of the experiment answered? Were all the stated requirements met?			17	
ADVANCED ANALYSIS		How aggressively did the report discuss the subtle relationships? How well were numerical relationships between variables developed and explained? How creative was the work presented via written text, graphs, tables and such like. (I really care about this folks!)			20	
NOTEBOOK		See Notebook Rubric – page 27			2	
PAPER		%	NOTEBOOK	/2	TOTAL	%

Signature Page

Please sign and indicate what each person did.

SIGNATURE	NAME	CONTRIBUTION
	Sarah Ali	programming theory introduction formatting
	Arthur Xu	programming theory data analysis formatting
	George Paraschiv	programming data analysis introduction formatting

2D COLLISION SIMULATION MARKING RUBRIC – APC2021

Student1 : _____ Student2 : _____ Student3 : _____

Student4 : _____ Student5 : _____

Regardless of the categories below, please submit this assignment using the following sections

1. Spring Analysis
2. Flow chart
3. Program Listing
4. 1D Collision example
5. 2D Collision example (fixed target)
6. 2D Collision example (moving target)

CATEGORY	DESCRIPTION	MARKS	GRADE
SPRING CONSTANT	Analysis of spring constant pursuant to accepted scientific techniques. This would include adequate uncertainty analysis of the data you collect in this area. The regressions should be performed according to course standards. The presentation of this materials should be effected using APC paper standards, including a description of the techniques used to obtain this relationship.	10	
SAMPLE PROBLEMS 1D 2D 2DM	Are these sample problems appropriate to the type of question required? Do they provide an adequate worked solution to allow you to compare with the performance of your simulation?	2+4+4	
PROGRAM LISTING	Program steps and routines should be extensively documented to allow an informed reader to easily understand the flow of your program. Special attention should be paid to the kernel of your computation loop so that the reader can easily perceive what your simulation is actually doing. In your documentation, there should be and indication of the area of the flow chart a particular subroutine represents. See requirements of flowcharts below.	20	
1D DEMO RESULT	<ol style="list-style-type: none"> 1. You are to include the required graphs for the two items in a single graph of x vs. t 2. You are to show ten lines of simulation output 3. You are to produce a simple table that shows the input and final values of the velocity vectors (1D in this case) of both the example question and your simulation. 4. Write a paragraph describing your choice for Δt and your stopping criteria 	14	

CATEGORY	DESCRIPTION	MARKS	GRADE
2D DEMO RESULT	<ol style="list-style-type: none"> 1. You are to include the required graphs for the two items in a single position graph of x vs. y 2. Your plots should trace the centres of mass of both objects. During contact your graph should include two greyed (i.e. dimmer) lines that show the non-compressed boundaries of both objects. No error bars are expected in a simulation as it is largely idealized. 3. You are to show ten lines of your simulators output in a table with appropriate headings. 4. You are to produce a simple table that shows the input and final values of the velocity vectors (2D in this case) of both the example question and your simulation and the differences between them (called residuals). If the residuals are more than 1% and explanation is required. 5. Write a paragraph describing your choice for your time step (Δt) and your criteria for ending your simulation. 	16	

CATEGORY	DESCRIPTION	MARKS	GRADE
2D MOVING TARGET RESULT	<p>1. You are to include the required graphs for the two items in a single position graph of x vs. y</p> <p>2. Your plots should trace the centres of mass of both objects. During contact your graph should include two greyed (i.e. dimmer) lines that show the non-compressed boundaries of both objects. No error bars are expected in a simulation as it is largely idealized.</p> <p>3. You are to show ten lines of your simulators output in a table with appropriate headings.</p> <p>4. You are to produce a simple table that shows the input and final values of the velocity vectors (2D in this case) of both the example question and your simulation and the differences between them (called residuals). If the residuals are more than 1% and explanation is required.</p> <p>5. Write a paragraph describing your choice for your time step (Δt) and your criteria for ending your simulation.</p>	20	
FLOW CHART	<p>Your flow chart should use standard symbols for this type of communication. In addition, your are required to put dashed lines around the symbols of a given subroutine. This box should be labelled with the subroutine name given in your program listing.</p> <p>If you decide to produce a series of flowcharts for subroutines and such like in your program, you are required to make a master flow chart demonstrating the overall organization of your simulation engine.</p>	10	
BONUS	<p>1. You must first write an explanation of how you approached the rotational problem. IN this you need also include if you are dealing with rotating targets and progenitors or just the rotation generated from the collision proper.</p> <p>2. The collision should be a 2D glancing collision with a stopped target as a minimum.</p> <p>3. You are probably going to require the use of an on-line simulator to be able to verify your results. You need to convince me that this simulator produces adequate results. I will leave it to your team to decide how impress me in this area.</p> <p>4. Angular momentum should be conserved in this situation, so you should demonstrate this aspect AS WELL as the linear results in your summary table.</p> <p>5. You should produce graphs of angular momentum showing the angular momentum of each item both as predicted by the on-line simulator and your own. On such a plot should also be the total system angular momentum computed as a sum of the angular momentum of each object. Such a plot should be horizontal in a plot of L vs. t</p>	20 max	
	TOTAL		%