7th International Conference on Microelectronics and Computer Science, Chişinău, Republic of Moldova, September 22-24, 2011

Implementation of physical impact laws, using Processing graphic language

Mihail KULEV, Vladislav LEDNIOV
Technical University of Moldova
vlad.ledniov@gmail.com , mkmk54@mail.ru

Abstract – The program is based on creating an algorithm that would process fluently an array- list with dynamically allocated memory, which works on an undefined number of objects, each of them serving for analyzing the movement of and the energy exchanges between each of them.

Index Terms – Array-list, elastic collision, energy, momentum, velocity.

1. INTRODUCTION

The purpose of the soft is to create a graphical equivalence of a vacuum-space environment, in order to see and study the elastic collisions between many objects in the action place. As objects are taken balls, of different sizes, with their own radius, velocity and weight (which corresponds to the radius).

2. TOPIC RESEARCH

The topic is related to mechanical laws, and in this thesis work, it is going to be reviewed the definition of the Elastic collision. An **elastic collision** is an encounter between two bodies in which the total kinetic energy of the two bodies after the encounter is equal to their total kinetic energy before the encounter. Elastic collisions occur only if there is no net conversion of kinetic energy into other forms.

Consider two particles, denoted by subscripts 1 and 2. Let m_i be the masses, u_i the velocities before collision and v_i the velocities after collision.

The conservation of the total momentum demands that the total momentum before the collision is the same as the total momentum after the collision, and is expressed by the equation:

$$m_1u_1 + m_2u_2 = m_1v_1 + m_2v_2. \quad (1)$$

Likewise, the conservation of the total kinetic energy is expressed by the equation:

$$\frac{m_1u_1^2}{2} + \frac{m_2u_2^2}{2} = \frac{m_1v_1^2}{2} + \frac{m_2v_2^2}{2}. \quad (2)$$

These equations may be solved directly to find v_i when u_i are known or vice versa. However, the algebra can get messy. A cleaner solution is to first change the frame of reference such that one of the known velocities is zero. The unknown velocities in the new frame of reference can then be determined and followed by a conversion back to the original frame of reference to reach the same result. Once one of the unknown velocities is determined, the other can be found by symmetry.

Solving these simultaneous equations (1) and (2) for v_i we get:

$$v_1 = \frac{u_1(m_1 - m_2) + 2m_2u_2}{m_1 + m_2}$$

$$v_2 = \frac{u_2(m_2 - m_1) + 2m_1u_1}{m_1 + m_2}$$

Equations (3)

Or, it can be obtained the following property:

$$v_1 = u_1, v_2 = u_2.$$

The latter is the trivial solution, corresponding to the case that no collision has taken place (yet).

Let's take an example:
Ball 1: mass = 3 kg, v = 4 m/s
Ball 2: mass = 5 kg, v = −6 m/s
After collision:
Ball 1: v = −8.5 m/s
Ball 2: v = 1.5 m/s
Property:

$$v_1 - v_2 = u_2 - u_1$$

Using the equation (2) we can write:

$$m_1\left(v_1^2 - u_1^2\right) = m_2\left(u_2^2 - v_2^2\right)$$
$$\Rightarrow m_1(v_1 - u_1)(v_1 + u_1) = m_2(u_2 - v_2)(u_2 + v_2)$$

Rearrange momentum equation:

$$m_1(v_1 - u_1) = m_2(u_2 - v_2)$$

Dividing kinetic energy equation by the momentum equation we get:

$$v_1 + u_1 = u_2 + v_2$$
$$\Rightarrow v_1 - v_2 = u_2 - u_1$$

the relative velocity of one particle with respect to the other is reversed by the collision the average of the momenta before and after the

As can be expected, the solution is invariant under adding a constant to all velocities, which is like using a frame of reference with constant translational velocity.

The velocity of the center of mass does not change by the collision:

The center of mass at time t

before the collision and at time t' after the collision is given by two equations:

$$\bar{x}(t) = \frac{m_1 \cdot x_1(t) + m_2 \cdot x_2(t)}{m_1 + m_2}$$

And

$$\bar{x}(t') = \frac{m_1 \cdot x_1(t') + m_2 \cdot x_2(t')}{m_1 + m_2}$$

Hence, the velocities of the center of mass before and after the collision are:

$$v_{\bar{x}} = \frac{m_1 u_1 + m_2 u_2}{m_1 + m_2}$$, and

$$v'_{\bar{x}} = \frac{m_1 v_1 + m_2 v_2}{m_1 + m_2}$$

The numerator of $v_{\bar{x}}$ is the total momentum before the collision, and numerator of $v'_{\bar{x}}$ is the total momentum after the collision. Since momentum is conserved, we have $v_{\bar{x}} = v'_{\bar{x}}$.

With respect to the center of mass both velocities are reversed by the collision: in the case of particles of different mass, a heavy particle moves slowly toward the center of mass, and bounces back with the same low speed, and a light particle moves fast toward the center of mass, and bounces back with the same high speed.

From the equations for v_1 and v_2 above we see that in the case of a large u_1, the value of v_1 is small if the masses are approximately the same: hitting a much lighter particle does not change the velocity much, hitting a much heavier particle causes the fast particle to bounce back with high speed.

3. USER INTERFACE

The program works as follows: firstly, a window appears with 2 balls bouncing in the window. The balls have their own weight and speed, and within the interaction, they interchange the speed by the mechanical laws which were reviewed in Part 1 of the thesis work. If the mouse is clicked within the window, it generates another ball on the mouse position, with some random weight and random speed.

The program analyses the movement of any 2 balls, and that may appear as a problem, because if there is a collision of more than 2 balls, they may clump with each other.

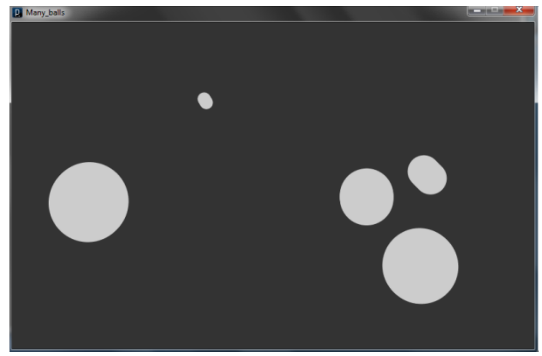

Fig.1 User Interface.

4. CONCLUSION

As a conclusion to the work, it can be said that any physical law can be implemented in a programming language, everything you need is the properties of the environment, with respect to any other forces or laws that interact with your workspace.

5. ACKNOWLEDGEMENT

The program was created thanks to the internet encyclopedia, Wikipedia.org, and special thanks to the open-source community, for sharing an open-source graphic programming language, processing.org.